

Webにエスパーを
求めるのは間違っているだろうか

CTF for ビギナーズ
前田 優人

れっくすです



- ・ 前田 優人
- ・ seccamp2012Web→2015チューター落ちました
- ・ CTF Team: dodododo, fuzzzi3
- ・ 最近の趣味はpwnの問題を解くこと
- ・ @xrekkusu

目次

- ・ 注意事項
- ・ 調査
- ・ Webはエスパー
- ・ SQLインジェクション
- ・ ディレクトリトラバーサル
- ・ Webいろいろ
- ・ おわりに

Web問に触れる前に
注意事項

注意事項

- ・ 一般のサイトに対して攻撃を行わないこと
- ・ 罪に問われる可能性があります
- ・ 攻撃をしたいときはCTFなどの専用ページで

バグバウンティ制度

- ・ 脆弱性を報告してくれた人に企業などが報奨金を支払う仕組み
- ・ Google, サイボウズ, HackerOne
- ・ 制度ごとにもルールがあるので要確認

まずは
調査

調査をしよう

- ・ Webの問題を解くときに必ず通る部分
- ・ 手がかりが何も見つからないと大変
- ・ 自分の知識や経験を全力でぶつける

調査に使う道具

- ・ デベロッパツール
- ・ Google
- ・ 脆弱性診断用プロキシツール

デベロッパ(開発者)ツール

- ・ ブラウザに標準で搭載されている管理者向けツール
- ・ DOMツリーや生HTTPを読むことができる
- ・ Web問を見たら真っ先に起動
- ・ Win: Ctrl-Shift-I (Firefox, Chrome)
- ・ Mac: Cmd-Option-I (Firefox, Chrome)

Filter

All | XHR | Script | Style | Images | Media | Fonts | Documents | WebS

Name	× Headers Preview Response Timing
<div>localhost</div>	<div>▼ General</div> <div>Remote Address: [::1]:8888</div> <div>Request URL: http://localhost:8888/</div> <div>Request Method: GET</div> <div>Status Code: ● 200 OK</div>
<div>favicon.ico</div>	<div>▼ Response Headers view parsed</div> <div>HTTP/1.1 200 OK</div> <div>Host: localhost:8888</div> <div>Connection: close</div> <div>X-Powered-By: PHP/5.6.9</div> <div>Content-type: text/html; charset=UTF-8</div>
	<div>▼ Request Headers view parsed</div> <div>GET / HTTP/1.1</div> <div>Host: localhost:8888</div> <div>Connection: keep-alive</div> <div>Cache-Control: max-age=0</div> <div>Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8</div> <div>User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_3) AppleWebKit/537.36 (KHTML</div> <div>Accept-Encoding: gzip, deflate, sdch</div> <div>Accept-Language: ja,en-US;q=0.8,en;q=0.6</div>

古かったら脆弱性がある？ → ぐるう

Filter

Name

localhos

favicon.

Host: localhost:8888

Connection: close

X-Powered-By: PHP/5.6.9

Content-type: text/html; charset=UTF-8

▼ Request Headers view parsed

HTTP/1.1 200 OK

Host: localhost:8888

Connection: close

X-Powered-By: PHP/5.6.9

Content-type: text/html; charset=UTF-8

▼ Request Headers view parsed

GET / HTTP/1.1

Host: localhost:8888

Connection: keep-alive

Cache-Control: max-age=0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/42.0.2311.91 Safari/537.36

Accept-Encoding: gzip, deflate, sdch

Accept-Language: ja,en-US;q=0.8,en;q=0.6

PHPのバージョン

🚫 🏠 <top frame> ▼ ☐ Preserve log

```
> var x = 10  
↩ undefined
```

```
> x  
↩ 10
```

```
> |
```

Google

- ・ 脆弱性チートシートを探す
- ・ 知らない脆弱性でもよく調べればだいたいなんとなかなる
- ・ 「いつもと何か違う？」と思ったらググっておこう

脆弱性診断用プロキシツール

- ・ BurpSuite / OWASP ZAP / Fiddler など
- ・ 通信のロギング
- ・ 通信の改ざん
- ・ 多機能すぎるので軽い紹介に留めます
- ・ 使いこなせると強力です

BurpSuite

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

Site map Scope

Filter: Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders

Host	Method	URL	Params	Status	Length	MIME type	Title
http://www.coins.tsuk...	GET	/		200	1485	HTML	
http://www.coins.tsuk...	GET	/favicon.ico		200	250		
http://www.coins.tsuk...	GET	/ce/				HTML	
http://www.coins.tsuk...	GET	/curri/				HTML	
http://www.coins.tsuk...	GET	/publicity/2015/sum...				HTML	
http://www.coins.tsuk...	GET	/shinro/				HTML	
http://www.coins.tsuk...	GET	/~internship/				HTML	

Request Response

Raw Headers Hex

GET / HTTP/1.1
Host: www.coins.tsukuba.ac.jp
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.10; rv:38.0)
Gecko/20100101 Firefox/38.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: keep-alive

? < + > Type a search term 0 matches

BurpSuite

Burp Intruder Repeater Window Help

Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts

Intercept HTTP history WebSockets history Options

Request to https://www.coins.tsukuba.ac.jp:443 [130.158.87.1]

Forward Drop Intercept is on Action

Raw Params Headers Hex

GET /enrolled_student/ HTTP/1.1
Host: www.coins.tsukuba.ac.jp
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.10; rv:38.0) Gecko/20100101 Firefox/38.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ja,en-US;q=0.7,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: https://www.coins.tsukuba.ac.jp/prep_student/
Cookie: __utma=185487291.444464324.1433918892.1433918892.1433918892.1; __utmb=185487291.2.10.1433918892;
__utmc=185487291; __utmz=185487291.1433918892.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none); __utmt=1
Connection: keep-alive

? < + > 0 matches

Webの基礎知識

HTTP

- ・ Webの基本
- ・ 直接書き換える機会があるので少しは知っておこう
- ・ おすすめ書籍 「HTTPの教科書」

HTTPリクエストヘッダ

GET / HTTP/1.1

Host: 2015.seccon.jp

Connection: keep-alive

Pragma: no-cache

Cache-Control: no-cache

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_3) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/44.0.2403.61 Safari/537.36

HTTPS: 1

Accept-Encoding: gzip, deflate, sdch

Accept-Language: ja,en-US;q=0.8,en;q=0.6

Cookie: _ga=GA1.2.2135735879.1433832162

HTTPレスポンスヘッダ

HTTP/1.1 200 OK

Date: Sun, 28 Jun 2015 12:21:21 GMT

Server: Apache/2.4.6 (CentOS)

Last-Modified: Mon, 22 Jun 2015 03:06:59 GMT

ETag: "286d-5191291320c6a"

Accept-Ranges: bytes

Content-Length: 10349

Keep-Alive: timeout=5, max=100

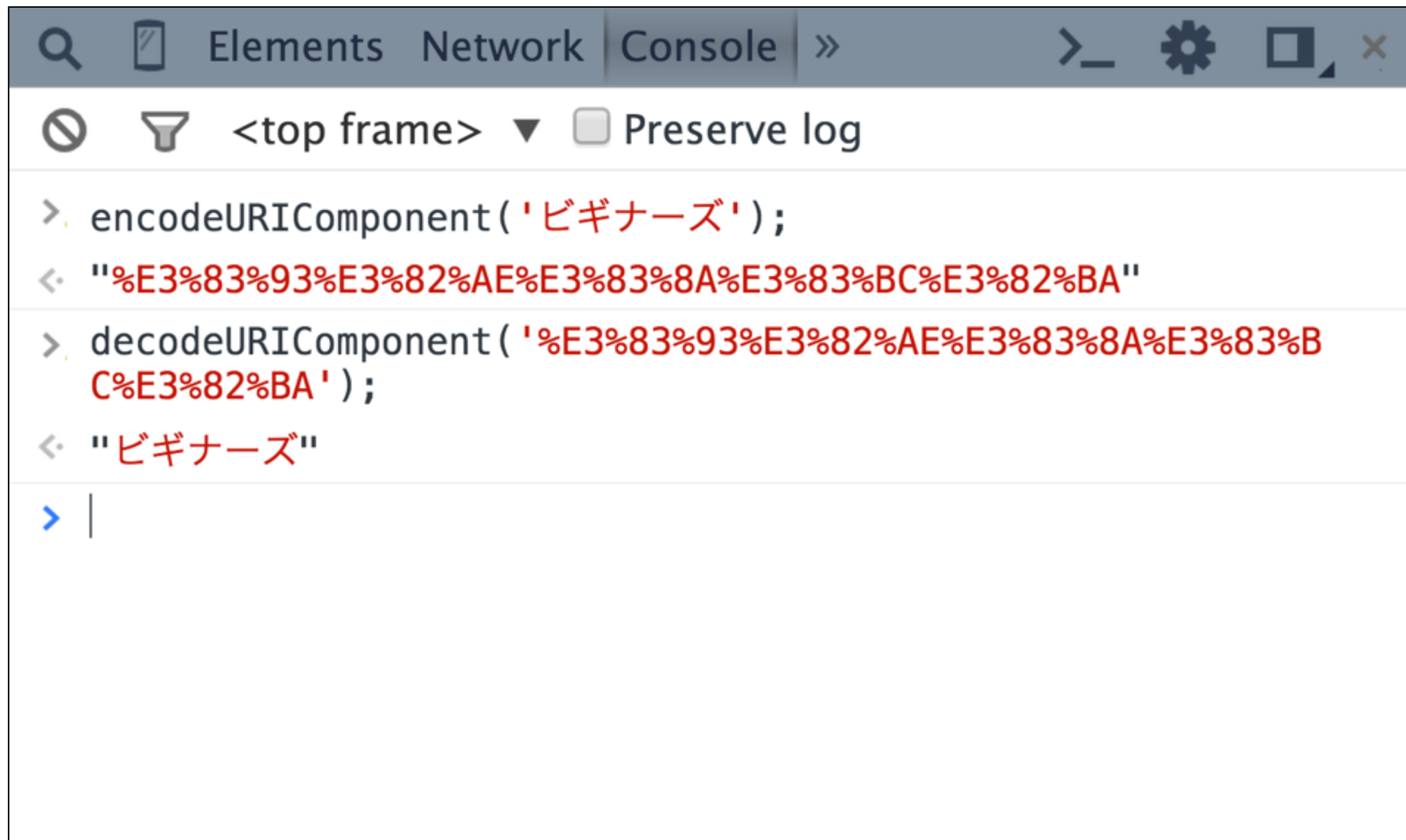
Connection: Keep-Alive

Content-Type: text/html; charset=UTF-8

パーセントエンコード

- ・ %3C%3E ← こういうやつ
- ・ %XX の形式でXXには16進ASCIIを入れる
- ・ HTTPで&とか#とか日本語を使いたいとき
- ・ `http://~/index.php?q=%3Cscript%3E`
- ・ 暗記してしまうか、 `encodeURIComponent`

パーセントエンコード



The screenshot shows a web browser's developer console with the 'Console' tab selected. The console displays the following JavaScript code and its output:

```
> encodeURIComponent( 'ビギナーズ' );  
< "%E3%83%93%E3%82%AE%E3%83%8A%E3%83%BC%E3%82%BA"  
  
> decodeURIComponent( '%E3%83%93%E3%82%AE%E3%83%8A%E3%83%BC%E3%82%BA' );  
< "ビギナーズ"  
  
> |
```

The console interface includes a toolbar at the top with icons for search, mobile view, and window management. Below the toolbar, there are tabs for 'Elements', 'Network', and 'Console'. The console also shows a filter icon, a dropdown menu set to '<top frame>', and a checkbox for 'Preserve log'.

Cookie

- ・ サーバーがユーザーのブラウザに一時的に情報を保存する仕組み
- ・ アクセスのたびに適切なCookieがサーバーに送信される
- ・ KEY=VALUEの形

Cookieの確認

Chromeではデベロッパツール→Resources→Cookies



	Name	Value	Domain	Path	Expires ...	Size	HTTP	Secure	First-Party
www.facebook.com	lu		.facebook.com	/	2017-0...	26	✓	✓	
	presence		.facebook.com	/	Session	185		✓	
	datr		.facebook.com	/	2017-0...	28	✓		
	s		.facebook.com	/	2015-0...	24	✓	✓	
	xs		.facebook.com	/	2015-0...	46	✓	✓	
	csm		.facebook.com	/	2015-0...	4			
	act		.facebook.com	/	Session	20			
	c_user		.facebook.com	/	2015-0...	21		✓	
	fr		.facebook.com	/	2015-0...	72	✓		
	p		.facebook.com	/	Session	3			

JavaScript

- ・ ECMAScript 6が承認されました
- ・ 出てきた時に読めるようにしましょう
- ・ 後述するXSSにおけるフィルタ回避など

調査のための攻撃

どんなことをする？

- ・ Webアプリが予想していないようなことをしよう
- ・ 例)

1. a.php?id=^{数 字}123 → a.php?id=^{文字・記号}A'B

2. 画像アップロードにPHPをアップロード

なにがおきる？

1. エラーメッセージ
2. 不審な挙動
 - ・ 変な文字列が出る
 - ・ 画像を受け付けるはずがPHPを受け付けた
3. 何の変哲もない普通の挙動

Warning: PDO::query(): SQLSTATE[HY000]: General error: 1
unrecognized token: "";" in **/home/ctf4b/sample/sqlerror.php**
on line **6**

試行→確認

- ・ Webはソースコードを与えられることが少ない
- ・ ヒントは**自分の手**で探し出す
- ・ エラーメッセージが出るのは簡単な部類
- ・ なんでも試すことがWebでは最も大切です

CTFで怪しい場所

怪しい場所を探す

- ・ URLパラメータ
- ・ ログイン画面
- ・ ファイルアップロード
- ・ 入力すると結果が表示される部分全般
- ・ **自分しか見られないページ**

自分しか見られないページ

- ・ Web問題は他のCTF参加者と同じのサービスを使う
- ・ 他人の解法を簡単に見られるようじゃだめ
- ・ 例えばデータ一覧ページは脆弱性を仕込みにくい

自分しか見られないページ

- ・ Web問題は他のCTF参加者と同じのサービスを使う
- ・ 他人の解法を簡単に見られるようじゃだめ
- ・ 例えばデータ一覧ページは脆弱性を仕込みにくい

Challenge

サーバーを用意しました

- ・ 接続先はこちら
- ・ `http://172.20.1.61/step1/`

実際の攻撃に入る前に
Webはエスパー

Webがエスパー？

- ・ バイナリ問題は逆アセンブルでロジックを追える
- ・ Web問題はソースコードすら提示されない場合が多い
- ・ 経験や勘が必要とされる局面がある

攻撃手法がいろいろ

- ・ バイナリ(Pwn)に対する攻撃手法は多くない
- ・ BufferOverflow/Use After Free/FormatStringAttackがほとんど
- ・ Webは攻撃手法が様々
 - ・ クロスサイトスクリプティング/SQLインジェクション
 - OSコマンドインジェクション/パラメータ改ざん
 - ディレクトリトラバーサル/任意コード実行/リプレイ攻撃/...

考察が大事

- ・ 調査した結果を元に考察を行う
 - ・ 「管理者画面に侵入する問題っぽい？」
 - ・ 「脆弱性はあるけどフィルタがあって面倒そう」
 - ・ 「この動きは問題と関係なさそうだ」
 - ・ 「あの人が作った問題はSQLiteだな」

それでもだめなとき

- ・ **Webはエスパー←重要**
- ・ .gitフォルダ にアクセスしてみたり
- ・ index.php~ (vimの一時ファイル) にアクセスしてみたり
- ・ もはや何でもあり
- ・ (泥沼化して誰も解かない問題が出ることも…)

いよいよ
攻撃

攻撃ですよ！

- ・ 脆弱性の紹介
- ・ 調査の様子
- ・ 攻撃方法
- ・ ここがエスパー

SQLインジェクション

SQLとは

- ・ SQLというデータベース問い合わせ言語
- ・ WebアプリからSQLを用いてDBに問い合わせをする
- ・ `SELECT id, pw FROM users WHERE user='admin';`



カラム名

テーブル名

条件

SQLインジェクションとは

- ・ 長いので SQLi と呼ばれる
- ・ WebアプリがSQLを生成するときのバグを利用
- ・ `SELECT * FROM users WHERE user='{ $user }';`
- ・ `$user` に `AAAA'AAAA` が入ってきたら……？
- ・ `SELECT * FROM users WHERE user='AAAA'エラーAAAA';`

SQLインジェクションの調査

- ・ シングルクオート・ダブルクオートを入れてみる
→ エラーは出る？
- ・ ' or '1 を入れてみる
- ・ ログインできてしまったり？

SQLインジェクション攻撃

- ・ `SELECT * FROM table WHERE user='{ $user }';`
- ・ `$user` に ' OR 1=1 -- を入れたら？
- ・ `SELECT * FROM table WHERE user=' ' OR 1=1 -- ';`
- ・ SQLとして正しい文になってしまった！
- ・ `user=' ' OR 1=1` は常に成り立つので条件が帳消し
- ・ `--` はコメントアウト。 `#` でコメントするDBもある

UNION句

- ・ フラグが隣のテーブルにあったら？
- ・ 条件を操作するだけでは足りない
- ・ 2つのテーブルの検索結果(SELECT結果)を結合

1	2	3
C	T	F
f	o	r
P	r	o

+

a	b	c
C	T	F
f	o	r
B	e	g
i	n	n
e	r	s

=

1	2	3
C	T	F
f	o	r
P	r	o
C	T	F
f	o	r
B	e	g
i	n	n
e	r	s

UNION句を使う

- `SELECT id, pw FROM users UNION SELECT id, pw FROM users2;`

単純な2つのテーブルの結合

- `SELECT id, pw FROM users
WHERE id = 'test' UNION SELECT 'admin', 'admin';`

SQLインジェクションでUNIONを使う

- `SELECT id, name, desc FROM items WHERE name LIKE
'%' UNION 0, 'flag', flag FROM flag --%;`

カラム数を合わせる

- ・ UNION句は検索結果の行を結合
- ・ カラム数が合わないとエラー
- ・ UNION SELECT 1,1,...,1
- ・ ORDER BY [カラム番号]
- ・ ORDER BY 3

テーブルやカラム名を知りたい

- ・ SQLiteでは

```
SELECT sql FROM sqlite_master;
```

テーブル作成時のSQLが表示される

- ・ MySQLでは

```
SELECT table_name FROM INFORMATION_SCHEMA.TABLES;
```

```
SELECT column_name FROM INFORMATION_SCHEMA.COLUMNS  
WHERE table_name = 'table';
```

ここがエスパー

- ・ 目には見えないSQL文を想像する
- ・ エラーメッセージが出ないと気づかないことも
- ・ 入力に対するフィルタが掛かっていることがある
(or, and, selectという文字列が消去される等)

さらなる高みへ

- ・ GROUP_CONCAT
- ・ LOAD_FILE
- ・ Blind SQL Injection
- ・ Error based SQL Injection
- ・ 興味があれば調べてみよう

Challenge

- ・ 接続先はこちら
- ・ <http://172.20.1.61/step2-1/>

ディレクトリトラバーサル

ディレクトリトラバーサルとは

- ・ ディレクトリを上層に向かって辿ってしまう
- ・ ファイルパスに `../` が混ざると危険
- ・ `index.php?id=about` → `./page/about.php` を表示
- ・ `index.php?id=../index`
→ `./page/../index.php` = `./index.php` を表示

ディレクトリトラバーサル調査

- ・ パラメータによって表示が変わるページ
- ・ index.php?id=**about**, index.php?id=**contact**
- ・ 書き換えて ../ を入れたらエラーが出る？

ディレクトリトラバーサル調査

- ・ 怪しかったら確実に存在するファイルを使う
- ・ `/etc/passwd`
- ・ `../../../../../../../../../../../../../../../../etc/passwd`
- ・ 勝手に末尾に `.php` などを付け足している場合は不可

ディレクトリトラバーサル攻撃

- ・ CTFでの使用例
- ・ `/etc/passwd` を読むとフラグが書いてある
- ・ `index.php` を読むとフラグが書いてある

設定ファイルを見る

- ・ サーバの設定ファイルを読んでみる
 - アプリケーション周りを見てもフラグがないとき
 - 解析するときのヒントとして
- ・ `/etc/apache2/httpd.conf`
- ・ 環境依存でパスが変わるので面倒

ここがエスパー

- ・ どのファイルを読めばいいかわかりにくい
- ・ ディレクトリ名を推測することがある
(`/var/www/example.com/index.php`)

Challenge

- ・ 接続先はこちら
- ・ <http://172.20.1.61/step2-2/>

Webいろいろ

Cookieの書き換え

- ・ Cookieを書き換えるとフラグが出る問題
- ・ 保存された `admin=false` を `admin=true`
- ・ Chrome EditThisCookieプラグイン

Cookie系の問題

- ・ Cookieに保存されたJSONを書き換える
- ・ Cookieに保存されたPHPのシリアライズデータを
書き換える

`a:2:{i:0;i:0;i:1;a:2:{i:0;i:1;i:1;i:2;}}` ← こういうの

クロスサイトスクリプティング

- ・ みんな大好きXSS
- ・ CTFではXSSしてCookieを盗むものが主流
- ・ 人気でも実はそんなに出題されない
- ・ 対策フィルタをくぐり抜けるテクニックが必要

アプリケーションの文字出力

- ?name=Hoge
 - <html><body>Hello, Hoge</body></html>
- ?name=Hoge<script>alert(1)</script>
 - <html><body>Hello, Hoge<script>alert(1)</script></body></html>



XSSの練習

- XSS game - <https://xss-game.appspot.com/>
- alert(1) to win - <http://escape.alf.nu/>
- prompt(1) to win - <http://prompt.ml/>

困った時は

- ・ いろいろ試しても何も成果が出ない
- ・ Webエスパーすぎ……

困ったときは

- ・ HTTPのヘッダを見よう
- ・ robots.txt, sitemap.xml, .git などを探そう
- ・ 諦めよう
 - ・ 誰も解いてない問題かもしれない

おわりに

CTFのWebは浅く広い

- ・ 紹介したものだけでなく常に情報収集をしよう
- ・ 「知っている」人は簡単に解ける問題が出る
- ・ 情報を吟味しちゃんと考察しよう

脆弱性の対策方法

- ・ 個々の脆弱性については長くなるので省きます
- ・ **ユーザーを信用しない**
- ・ **データが入出力に気をつける**

CTFで強くなるために

- ・ 好き嫌いをしない(バイナリもやろう)
- ・ チームを組もう(2~4人程度がおすすめ)
- ・ Writeupを読もう

おすすめ書籍

- ・ 体系的に学ぶ 安全なWebアプリケーションの作り方
 - ・ 徳丸本と呼ばれていて、開発者向けの脆弱性対策について解説されています
- ・ めんどくさいWebセキュリティ
 - ・ Webの細かい「めんどくさい」部分の話を知りたい方は是非
- ・ Web Security Testing Cookbook
 - ・ 英語ですがツールの使い方から脆弱性の調べ方まで詳しく載っています