Leibniz Universität Hannover

Sicherheit von Webanwendungen

Wibke Börger

OWASP

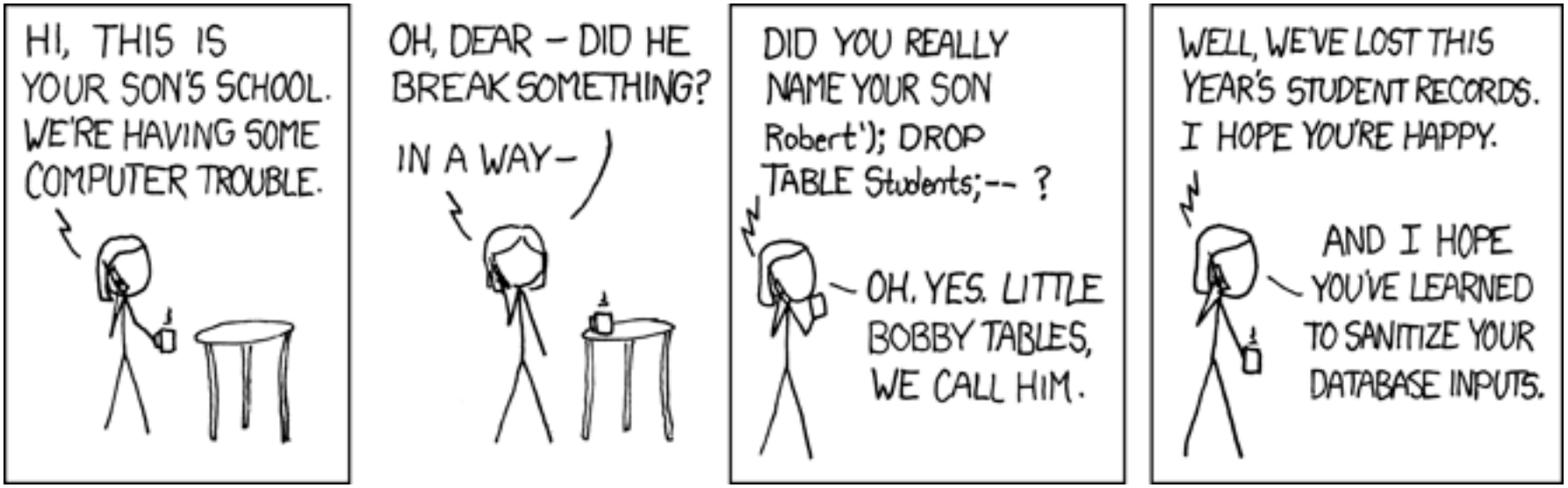
- Open Web Application Security Project
  - https://www.owasp.org/
- Organisation mit dem Ziel, die Sicherheit von Webanwendung zu verbessern
- viele Projekte, z.B.
  - Tools zum Testen von Webanwendungen
  - Bibliotheken für verschiedene Programmiersprachen

Leibniz Universität Hannover

OWASP Top Ten 2010

- Injection
- Cross-Site Scripting (XSS)
- Fehler in Authentifizierung und Session-Management
- Unsichere direkte Objektreferenzen
- Cross-Site Request Forgery (CSRF)
- Sicherheitsrelevante Fehlkonfiguration
- Kryptografisch unsichere Speicherung
- Mangelhafter URL-Zugriffsschutz
- Unzureichende Absicherung der Transportschicht
- Ungeprüfte Um- und Weiterleitungen

SQL Injection



http://xkcd.com/327/

SQL Injection - Beispiel

- Typische Login-Seite:

SQL Injection - Beispiel

- SQL-Abfrage:

```
$sql = "select * from users where
username='$username' and
password='$password'";
```

SQL Injection - Beispiel

- SQL-Abfrage:

```
$sql = "select * from users where
username='$username' and
password='$password'";
```

- Mit normaler Benutzereingabe:

```
$sql = "select * from users where
username='admin' and password='geheim'";
```

SQL Injection - Beispiel

- SQL-Abfrage:

```
$sql = "select * from users where
username='$username' and
password='$password'";
```

- SQL-Injection:

```
$sql = "select * from users where
username='admin' and password='' or '1'='1'";
```

SQL Injection

Demonstration

SQL Injection - Gegenmaßnahmen

- Prepared Statements verwenden

```
$db = new PDO("host=$dbhost;dbname=$dbname",
$dbuser, $dbpasswd);

$sth = $db->prepare('select * from users where
username=? and password=?');

$sth->execute(array($username, $password));
```

Cross Site Scripting (XSS)

- Injection, meist Javascript
- wird im Browser des Nutzers ausgeführt

- ermöglicht z.B.:
  - Defacement
  - Übernahme von Sessions

Cross Site Scripting (XSS)

Demonstration

XSS - Gegenmaßnahmen

- XSS zu filtern ist schwierig, da es eine Fülle von Variationen gibt:
  - https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet
- Toolkits/Libraries verwenden:
  - OWASP Enterprise Security API
    - https://www.owasp.org/index.php/ESAPI
  - HTML Purifier
    - http://htmlpurifier.org/
- Cookie/Session-Diebstahl verhindern:
  - HTTPOnly Cookies
  - Sessions an IP binden

Cross Site Request Forgery (CSRF)

- Angreifer führt Aktionen im Namen des Opfers aus

CSRF - Beispiel

- Herr L. kauft häufig in einem Internetshop ein
- der Internetshop hat ein 1-Click-Buy-Feature, das über folgenden Link aufgerufen wird
  - http://example.com/buy?item=ARTIKEL&qty=ANZAHL
- Herr L. besucht ebenso häufig ein Internetforum
- der Angreifer kennt die Surfgewohnheiten von Herrn L., und platziert im Forum folgenden Code
  -

CSRF - Beispiel

- Herr L. loggt sich im Internetshop ein, läßt die Session geöffnet und besucht wenig später das Forum
- im Forum ruft Herr L. die vom Angreifer modifizierte Seite auf, und sein Browser lädt automatisch die URL des Internetshops
- da Herr L. im Internetshop eingeloggt ist, schickt der Browser das Cookie mit der Sessionid mit
- der Internetshop erhält die Bestellung und ordnet sie aufgrund der Sessionid Herrn L. zu
- Herr L. hat 150 Gläser Senf gekauft

CSRF

Demonstration

CSRF - Gegenmaßnahmen

- Tokens
  - jedes Formular erhält ein dynamisch generiertes Token, das mit der Session des Benutzers verknüpft ist
  - nur wenn das mit dem Formular mitgeschickte Token mit dem zugehörigen Session-Token übereinstimmt, wird die Anfrage ausgeführt