Westfälische Wilhelms-Universität Münster

Erstellen dynamischer Webseiten mit PHP

WS 2016/2017

wissen.leben
WWU Münster

Dozenten: Patrick Förster, Michael Hasseler

Übersicht der heutigen Inhalte

13. Vorlesung: Angriffsarten bei PHP-Skripten
- Cross-Site-Scripting
- Cross-Site-Request-Forgeries
- SQL-Injection
- Shell-Injection

Cross-Site-Scripting

- „XSS“: Drittcodeausführung
- Angreifer übermittelt eigenen HTML-Code an eine Webanwendung
- Webanwendung speichert den fremden HTML-Code unmaskiert z. B. als folgenden Gästebucheintrag: (b01\index.php)

```
<script type="text/javascript">
      location.href="http://www.uni-muenster.de"
 </script>
```

- Abhilfe: **`htmlspecialchars()`** bzw. **`htmlentities()`**
- **`strip_tags`** hilft beim gezielten Entfernen von HTML.

Vgl. PHP5 Fortgeschrittene Techniken der Web-Programmierung, RRZN, 2.Aufl., Seite 180-182

Cross-Site-Request-Forgeries

- CSRF („c-surf“): Drittanforderungsfälschung
- Angriff ähnlich zu XSS, hier wird Fremdcode/http-Requests über das Einbinden von Objekten z. B. Bilder untergeschoben
- Cookies können ausgelesen werden
- Bestellungen können wg. vorhandener Cookies automatisch ausgeführt werden.
- Abhilfe: Einbinden von Bildern aus fremden Quellen nicht auf eigener Seite zulassen
- Beispiel:
- `<img src=“http://angreifer-seite.de/skript.php“>`

Vgl. PHP5 Fortgeschrittene Techniken der Web-Programmierung, RRZN, 2.Aufl., Seite 182-184

SQL-Injection

- SQL-Code-Einfügung
- Abhilfe:
  - **`addslashes()`** `oder` **`mysql_real_escape_string()`**
    (NICHT beides, sonst doppelt!)
  - PDO mit prepared Statements und bindingParams verwenden
  - GET gefährlicher als POST

Vgl. PHP5 Fortgeschrittene Techniken der Web-Programmierung, RRZN, 2.Aufl., Seite 184-187

Beispiel für SQL-Injection

- Eingabe von **paulchen** und **rosa123**:
  - `login.php?benutzer=paulchen&passwort=rosa123`
  - `SELECT COUNT(*) FROM benutzer WHERE login = 'paulchen' AND passwort = 'rosa123'`
  - Anmeldung erfolgreich, da Nutzerkennung und Passwort korrekt
- Eingabe von **paulchen** und **Abc ' OR ' 1 ' = ' 1** :
  - `login.php?benutzer=paulchen&passwort=Abc' OR ' 1 ' = ' 1`
  - `SELECT COUNT(*) FROM benutzer WHERE login = 'paulchen' AND passwort = 'Abc ' OR ' 1 ' = ' 1 '`
  - Anmeldung erfolgreich, obwohl Passwort falsch ist (SQL-Injection)

Vgl. PHP5 Fortgeschrittene Techniken der Web-Programmierung, RRZN, 2.Aufl., Seite 184-187

SQL-Injection verhindern (Beispiel 2)

- Testfälle bei unserem Fussballbeispielskript start.php durchführen:
  a. Mit PDO, prepared Statement und bindingParam für das Feld passwort
  b. Mit PDO, prepared Statement und kein bindingParam für das Feld passwort
  c. Mit PDO, prepared Statement und kein bindingParam für das Feld passwort, aber mit addSlashes() für die Eingabe des Passwort
- Ergebnis: SQL-Injection im Fall a) und c) nicht möglich
- Inhalte aus Eingabefeldern bei Verwendung von bindingParam vom PDO-Treiber maskieren lassen oder selbst mit addSlashes() maskieren
  - ➔ Nutzereingaben können kein eigenes SQL einschleusen
  - ➔ SQL-Injection wird verhindert

Quelle: vgl. http://www.php.net/pdo.prepared-statements

Shell-Injections

- Shell-Code-Einfügung
- Geht nur, wenn man Systemkommandos ausführt, die Nutzereingaben enthalten.
- Mit der Funktion shell_exec können Systemkommandos ausgeführt werden
- hinter **`&`** beginnt ein neuer Befehl bei Windows
- **`escapeshellcmd()`** anwenden um Shell-Injections zu verhindern!

Vgl. PHP5 Fortgeschrittene Techniken der Web-Programmierung, RRZN, 2.Aufl., Seite 187-190

Fertige Lösungen

- Nutzung von fertigen Lösungen aus dem Internet z. B. Gästebuch
  - die PHP-Dateien sind allen bekannt
  - können Schwachstellen und Fehler enthalten
  - regelmäßig auf Bugfixes prüfen
- Wenn die fertige Lösung nicht mehr gewartet wird, dann auf eine Alternative umsteigen.

Vgl. PHP5 Fortgeschrittene Techniken der Web-Programmierung, RRZN, 2.Aufl., Seite 172-173

Zusammenfassung

- Sie kennen die folgenden 4 Angriffsarten, können diese beschreiben und wissen wie man sie verhindert:
  - Cross-Site-Scripting
  - Cross-Site-Request-Forgeries
  - SQL-Injection
  - Shell-Injection

Anhang

- Um die Passwörter im Fußballbeispiel von Klartext in der Datenbank auf den SHA1-Hash des Passwortes zu ändern, führen Sie folgende zwei SQL-Statements aus:

  - `ALTER TABLE benutzer MODIFY COLUMN passwort VARCHAR(40) NOT NULL DEFAULT '';`

  - `UPDATE benutzer SET passwort = SHA1(passwort);`

- In der start.php muss dann folgende Zeile eingefügt werden:

  - $passwort = sha1($passwort);

  - Damit wird der sha1-Hash von eingegeben Passwort mit dem sha1 in der Datenbank verglichen