sirius-net GmbH

Penetrationstest ... und was dann?

Unternehmensvorstellung

Dynamik seit 2001

Zurzeit besteht unser Team aus 22 Mitarbeitern.

Gestartet als reines IBM Notes/Domino Entwicklerteam, beraten und unterstützen wir heute unsere internationalen Kunden in unterschiedlichsten Bereichen der Anwendungsentwicklung, der Datenpflege und -qualität sowie dem Einsatz unterschiedlicher Kommunikationssysteme.

Unser Portfolio umfasst hierbei die gesamte Bandbreite von der ersten Konzeption über die Umsetzung bis zum Datacenter-Betrieb.

Als IBM Silver Business Partner verfügt sirius-net über 25 Jahre umfassende Kompetenz im Portfolio der IBM Collaboration Solutions.

Der Schwerpunkt unserer Dienstleistungen liegt auf der Konzeption, der Entwicklung und dem Betrieb von komplexen Business Anwendungen und Schnittstellen.

Development & Digital Experience

Wenn wir entwickeln, bleiben wir gern in Kontakt. In enger Abstimmung mit dem Auftraggeber definieren wir ein Lastenheft und programmieren schon im frühen Stadium Prototypen, anhand derer gemeinsam über die weiteren Schritte entschieden wird.

IBM Domino

Websites, XPages, Classic Notes, PlugIns

In Sachen IBM Domino und Notes macht uns so leicht niemand etwas vor.

Schnittstellen

Connectivity
On Premises und als Cloud Service

Einen Moment bitte, wir verbinden.

PHP

Server Side Scripting für Webapplikationen.

Speziell auf Webapplikationen ausgerichtete Script-Sprache.

Typo3 & WordPress Open Source CMS

"Da brauchen wir niemanden, das können wir doch selbst!"

Penetrationstest

- Prüfung auf Sicherheitslücken
    - Infrastruktur
    - Webapplikation
- Sicherheitstest findet in einer abgesprochenen Zeitspanne (mehrere Tage / Wochen) statt
- Absprache der zu testenden Systeme
- Falls nötig, werden Test-Benutzer benutzt
- Hosting-Partner muss über Penetrationstest informiert werden

SQL-Injection

Werden eingegebene Befehle nicht entsprechend geprüft, kann jede Art von Code zur Ausführung kommen. Dieser Prozess wird SQL-Injection (SQL-Einschleusung) bezeichnet.

SQL-Statement:

```
$stmt = "SELECT COUNT(*)" .
        " FROM users " .
        " WHERE email = '" . $_POST['email'] .
        " AND password = '" . md5($_POST['pass']) . "';";
```

```
SELECT COUNT(*) FROM users WHERE email = 'x' OR
1=(DROP TABLE tabname); -- ' AND password =
'f6a84d2f…ab';
```

SQL-Injection

Werden eingegebene Befehle nicht entsprechend geprüft, kann jede Art von Code zur Ausführung kommen. Dieser Prozess wird SQL-Injection (SQL-Einschleusung) bezeichnet.

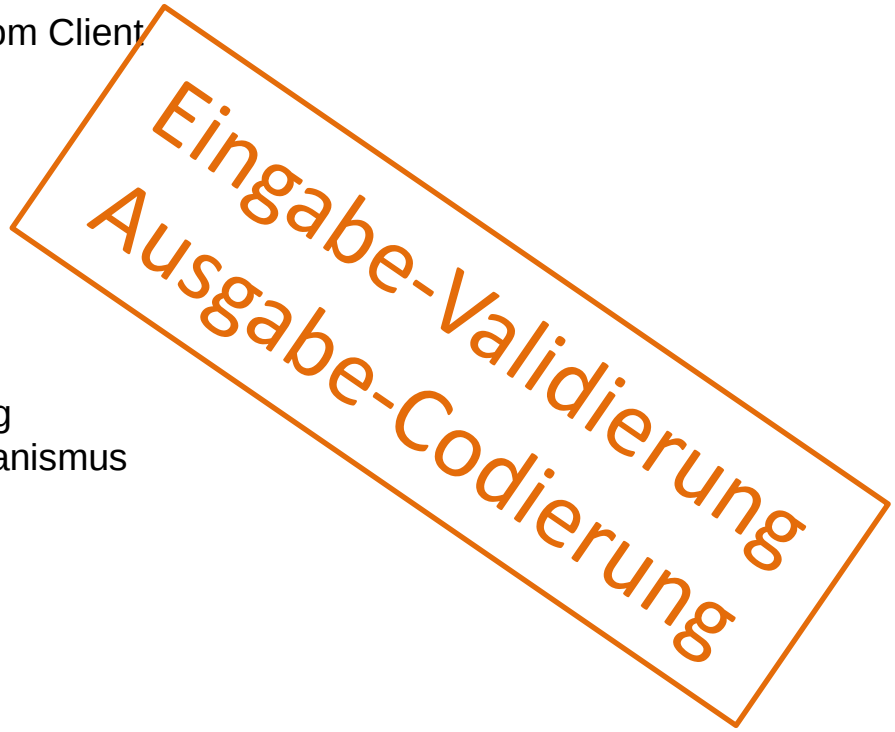
Lösungsmöglichkeit

- ' escapen
- Framework nutzen (Prepared Statements)

Cross-Site Scripting-Angriffe (XSS)

Derartige Angriffe erlauben es einem Angreifer beispielsweise, beliebigen JavaScript-Code im Kontext anderer Benutzer auszuführen.

- serverseitig umgesetzte Überprüfung aller vom Client übermittelten Daten
  - Formularfelder
  - URL-Parameter
  - Cookie-Inhalte
  - HTTP-Header
- Daten sollten vor einer erneuten Auslieferung nochmals einen entsprechenden Filtermechanismus durchlaufen.



Cross-Site Scripting-Angriffe (XSS)

Problemstellen:

- Textareas mit HTML-Eingabemöglichkeit
- WYSIWYG-Editoren
- Übergabe von Texten per GET-Parameter, z.B. Dateinamen

Möglichkeiten:

- Encoding von <, >, "
- Shortcodes anstelle der Eingabe von kritischen Tags
    - <script>, <iframe>, <style>, <svg>, <body>, <form>, <input>, <image>, <applet>, <embed>, <object>, etc
    - Tags können beliebige Attribute beinhalten, die ausführbaren JavaScript-Code enthalten

Cookies

Sämtliche Cookies eines Benutzers werden automatisch in jede HTTP-Anfrage eingebunden

- Das Fehlen des Attributes `HttpOnly` erlaubt es via JavaScript auf den Session-Identifier zuzugreifen
- Cross-Site Request Forgery (CSRF/XSRF)
- Cookies ermöglichen also Authentification-Hacking

Lösungsmöglichkeiten

- Attribut `HttpOnly`
- Pfad-Gültigkeit von Cookies einschränken
- Absicherung über einzigartige Tokens: sollte der Token des Clients nicht mit dem Server-Token übereinstimmen, wird die Aktion einfach verworfen

Verhinderung von (Spam-)Bots durch CAPTCHA

Spam-Bots aufzuhalten ist immer ein Rennen gegen die Zeit / den Fortschritt, die Programme können mehr lesen als man manchmal denkt.

Probleme:

- nicht jeder weiß mit CAPTCHAs umzugehen
- CAPTCHAs bestechen nicht immer durch Lesbarkeit

Alternative Möglichkeiten:

- Nachgelagertes Nachladen von Tokens o.ä.
- Prüfsumme der übertragenen Daten (o. ä.)
- "ViewID"

Schwachstellen beim Login

Fehlermeldung, die Aufschluss über die Existenz eines Benutzernamens geben, vereinfachen Brute-Force Attacken

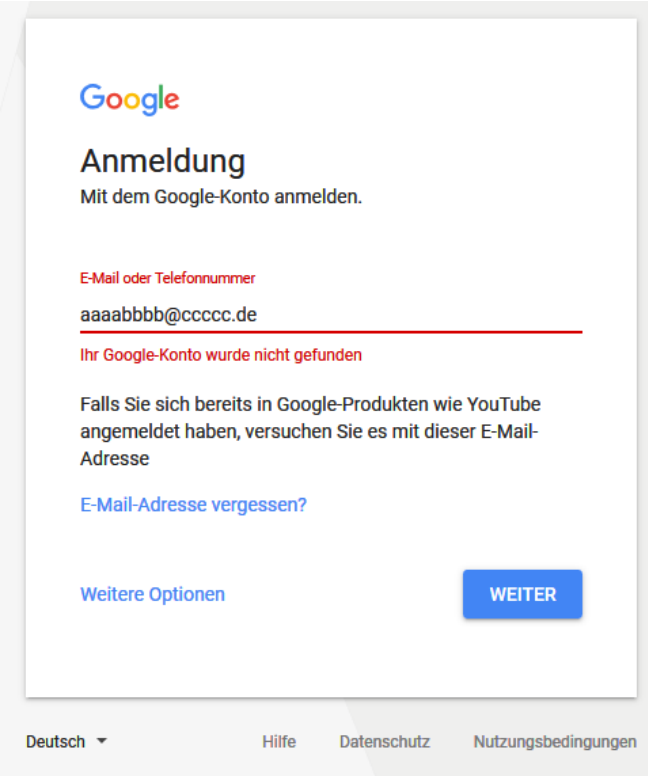
Unterschiedliche Verarbeitungszeiten von gefundenen vs. nicht gefundenen Benutzernamen gibt ebenfalls Aufschlüsse über die Existenz eines Benutzers

Lösung:

- Neutrale, oder gar keine Fehlermeldung
- Sperren eines Accounts für eine gewisse Zeitspanne, wenn zu viele Fehlversuche auflaufen
- Identische Verarbeitungszeiten

Problem:

- Der Benutzer erhält keine Fehlererklärung, er weiß nicht, ob Benutzername oder Kennwort falsch ist
- Bei Systemen mit temporären Benutzeraccounts kann dies zu Usability-Einbußen führen



Schwachstellen bei Registrierung

Eindeutigkeit von Angaben, zumeist Benutzername

Wird der Benutzer darauf hingewiesen, dass z.B. die angegebene e-Mail-Adresse bereits verwendet wird, erhält der Angreifer Informationen über die Existenz des entsprechenden Benutzers.

Bei Systemen, die Benutzerkonten direkt bei Registrierung anlegen, ist eine Prüfung auf Eindeutigkeit jedoch zwingend erforderlich.

Lösungsmöglichkeiten

- Menschen müssen aktiv Registrierungen prüfen und Benutzerkonten anlegen (skaliert nicht gut)
- Registrierung abschicken, bei bereits bestehendem Konto den Benutzer über die Sachlage informieren

Passwort-vergessen-Funktion

Fehlermeldung, die Aufschluss über die Existenz eines Benutzernamens geben, vereinfachen Brute-Force Attacken

Lösung:

- keine Fehlermeldung

  → Benutzer erfährt es nicht, wenn er gar kein Konto im System hat (z.B. bei Systemen mit temporär gültigen Benutzern)

- Immer Mail an den gefundenen Benutzer senden, mit einem entsprechendem Hinweis, wenn nicht registriert

  → Könnte für Spam-Zwecke benutzt werden, sollte also keine Massenversendung zulassen, nur eine Hinweismail pro Tag o. ä.

sirius-net GmbH
Marienstraße 43
D-70178 Stuttgart

Telefon +49 711 / 96 69 - 760
Telefax +49 711 / 96 69 - 770

info@siriusonline.de