Owasp Security Shepherd Project

Tarik EL AOUADI

Owasp France Meeting – Jul 7th 2016
tarik.elaouadi@owasp.org

OWASP
The Open Web Application Security Project
• Tarik EL AOUADI (tarik.elaouadi@owasp.org)
  – +5 years of software development
    • Accenture, Bull, ...
  – +5 years of software security
    • Consulting, code review, pentesting, training
  – Owasp Morocco Chapter Co-Leader
    • Speaker & Training Manager
    • Owasp Morocco Cyber Security Conference 2016
      – Marrakech (Nov 25th, 26th)
  – Entrepreneur (Adam Ridson)
    • Improving Software Security
The OWASP Security Shepherd project is a web and mobile application security training platform. Security Shepherd has been designed to foster and improve security awareness among a varied skill-set demographic. The aim of this project is to take AppSec novices or experienced engineers and sharpen their penetration testing skillset to security expert status.
The Security Shepherd project covers the following web and mobile application security topics

- SQL Injection
- Broken Authentication and Session Management
- Cross Site Scripting
- Insecure Direct Object Reference
- Security Misconfiguration
- Sensitive Data Exposure
- Missing Function Level Access Control
- Cross Site Request Forgery
- Unvalidated Redirects and Forwards
- Poor Data Validation
- Insecure Data Storage
- Unintended Data Leakage
- Poor Authentication and Authorisation
- Broken crypto
- Client Side Injection
- Lack Of Binary Protections
SQL Injection Lesson

Injection flaws, such as SQL injection, occur when hostile data is sent to an interpreter as part of a command or query. The hostile data can trick the interpreter into executing unintended commands or accessing unauthorized data. Injection flaws are of a high severity. Injection flaws can be exploited to remove a system's confidentiality by accessing any information held on the system. These security risks can then be extended to execute updates to existing data affecting the system's integrity and availability. These attacks are easily exploitable as they can be initiated by anyone who can interact with the system through any data they pass to the application.

The following form's parameters are concatenated to a string that will be passed to a SQL server. This means that the data can be interpreted as part of the code.

The objective here is to modify the result of the query with SQL injection so that all of the table's rows are returned. This means you want to change the boolean result of the query's WHERE clause to return true for every row in the table. The easiest way to ensure the boolean result is always true is to inject a boolean OR operator followed by a true statement like 1 = 1.

If the parameter is being interpreted as a string, you can escape the string with an apostrophe. That means that everything after the apostrophe will be interpreted as SQL code.

Exploit the SQL injection flaw in the following example to retrieve all of the rows in the table. The lesson's solution key will be found in one of these rows! The results will be posted beneath the search form.

Please enter the user name of the user that you want to look up

Get this user!
<table>
<thead>
<tr>
<th>Rank</th>
<th>Team</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st</td>
<td>dcua</td>
<td>3941</td>
</tr>
<tr>
<td>2nd</td>
<td>NULL Life</td>
<td>3558</td>
</tr>
<tr>
<td>3rd</td>
<td>arusell</td>
<td>3053</td>
</tr>
<tr>
<td>4th</td>
<td>andro1de</td>
<td>2996</td>
</tr>
<tr>
<td>5th</td>
<td>micaman</td>
<td>2966</td>
</tr>
<tr>
<td>6th</td>
<td>Insanity</td>
<td>2909</td>
</tr>
<tr>
<td>7th</td>
<td>longerthan5characters</td>
<td>2878</td>
</tr>
<tr>
<td>8th</td>
<td>aiacobelli</td>
<td>2516</td>
</tr>
<tr>
<td>9th</td>
<td>ottucsakj</td>
<td>2501</td>
</tr>
<tr>
<td>10th</td>
<td>mfocuz</td>
<td>2084</td>
</tr>
</tbody>
</table>
什么是不安全的直接对象引用？

想象一个允许您查看您的个人信息的网页。在该页面上，您的信息是产生了属于用户的 ID。如果这个页面是存在不安全的直接对象引用，则可能发生黑客获取您的个人信息并攻击您。不安全的直接对象引用是当一个应用程序设计时的一个错误。如果引用的对象是未安全引用，那么该对象可能会被攻击者利用，从而导致信息泄露。

攻击者可以利用不安全的直接对象引用来获取任何可以被引用的对象所引用的任何信息。在上面的例子中，攻击者可以访问任何用户的个人信息。

不安全的直接对象引用的严重程度取决于被攻击对象所引用的影响。如果这种情况发生，一个公司可能会丢失自己竞争对手的信息。这时，该对象对商业的影响是巨大的。如果该对象是必须被引用，并且不应该出现在专业级的程序中，

完成初始化的直接对象存在管理员的个人信息中

用户：游客

<table>
<thead>
<tr>
<th>年龄</th>
<th>22</th>
</tr>
</thead>
<tbody>
<tr>
<td>住址</td>
<td>54 Kevin Street, Dublin</td>
</tr>
<tr>
<td>邮箱</td>
<td><a href="mailto:guestAccount@securityShepherd.com">guestAccount@securityShepherd.com</a></td>
</tr>
<tr>
<td>个人信息:</td>
<td>没有个人信息设置</td>
</tr>
</tbody>
</table>
• https://www.owasp.org/index.php/OWASP_Security_Shepherd

• https://github.com/OWASP/SecurityShepherd

• https://github.com/OWASP/SecurityShepherd/releases/tag/v3.0v (latest release)
Connexion

Utilisez vos informations d'identification Security Shepherd pour vous connecter.

Enregistrer un compte Security Shepherd ici !

Nom d'Utilisateur: 

Mot de Passe: 

Soumettre

Avez-vous besoin d'un proxy ?
A Propos de Security Shepherd