Lab 1: Web application Security

Part 1. Juice Shop Schwachstellen finden

Zu Beginn wurden die erforderlichen Tools Schritt für Schritt installiert. Nachdem die Einrichtung abgeschlossen war, wurde anschließend die Juice Shop Webseite analysiert. Es wurden verschiedene Produkte ausgewählt und die einzelnen Menüpunkte untersucht. Im Anschluss wurden dann die erste Laboraufgabe gestartet. Die Aufgabe war es das Score Board zu finden und anschließend 3 Schwachstellen der Seite zu finden.

Die Suche nach dem Scoreboard:

Beim Durchsuchen der Webseite sind uns verschiedene Unterpunkte aufgefallen. Im Menü haben wir den Punkt „Help getting started“ gefunden. Mit einem Klick auf diesen Menüpunkt hat sich auch schon eine Hilfestellung geöffnet, wie man das Score Board finden kann. Es wurde empfohlen das DevTool mit F12 zu öffnen. Man konnte aber auch durch raten der URL das Score Board finden. Wir ahnten schon was man in der URL ergänzen mussten, aber wollten trotzdem den ausführlichen Weg gehen. Also öffneten wir mit F12 das Devtool. In dem Reiter „Sources“ konnte man verschiedene Javascript Dateien finden und und mit STRG+F die jeweilige Datei durchsuchen. Beim Durchsuchen der „main.js“ Datei mit dem Schlagwort „score“ sind wir beim zwölften Suchergebnis fündig geworden. Hier stand der Punkt (path:"score-board") der uns auf die URL-Lösung brachte. Nach ausprobieren in der URL-Zeile sind wir schließlich auch auf das Scoreboard gestoßen. In der Menüstruktur war nun auch das Scoreboard sichtbar.

Datenschutzerklärung finden:

Der nächste Schritt bestand darin eine weitere 1-Sterne und zwei 2-Sterne Aufgaben des Scoreboards zu lösen.  
Als 1-Stern Aufgabe wurde die Datenschutzbestimmung des Juice Shops gesucht. Auch hier bekam man wieder eine kleine Hilfestellung. Um die Bestimmungen zu finden wurde zunächst ein neuer Benutzer mit Name und Passwort erstellt. Hier haben wir als Test eine Emailadresse eingegeben mit einem Anbieter, der gar nicht existiert (z.B. @blabla.com). Das hat sogar funktioniert.

Nachdem der Benutzer erstellt wurde, konnte man die Datenschutzbestimmungen oben rechts über Konto – Datenschutz & Sicherheit finden und lesen.

Passwortstärke und Adminkonto ermitteln:

Als zwei 2-Sterne Aufgaben wurde die Ermittlung des Admin Passwort- und Benutzername ausgesucht. Für das Ausfindig machen des Benutzernamens wurden die Reviews der einzelnen Produkte untersucht. Schon bereits beim ersten Produkt gab es ein Kommentar des Admins, welchen man anhand der Emailadresse „[admin@juice-sh.op](mailto:admin@juice-sh.op)“ erahnen konnte. Beim Versuch sich mit dieser Emailadresse anzumelden fehlte jetzt nur noch das Passwort. Bei der Passworteingabe erschien ein Hinweis, dass ein sehr einfaches Passwort gewählt wurde. Also versuchten wir zunächst Zahlenfolgen wie 12345 oder 00000. Nachdem es mit der Zahlenfolge nicht funktioniert hat, versuchten wir es weiter mit Text und Zahlen. Schließlich gelang es uns mit dem Passwort „admin123“ uns in das Profil einzuloggen.

Part 2: Webanwendung entwickeln und Schwachstellen integrieren

Wir haben uns dazu entschieden eine Webanwendung für Notizen zu erstellen. Für die Anwendung soll es ein Login geben womit sich der Benutzer einloggen kann. Nachdem er sich eingeloggt hat werden ihm auch schon die ganzen von sich erstellten Notizen angezeigt. Von hier aus hat er auch die Möglichkeit direkt eine neue Notiz zu erstellen.

Die Benutzerdaten und Notizen wurden in einer MYSQL Datenbank gespeichert. Damit die Anwendung auch im Browser angezeigt wird, wurde in Javascript ein Node.js Server erstellt der mithilfe des Frameworks express.js den Austausch ermöglicht.

SQL Injection als erste Schwachstelle.

...

XSS Attacke

Für die Übermittlung von Backend Variablen an das Frontend haben wir in express die response.render() Funktion verwendet. Diese Funktion wird oft in Kombination mit dem Dateiformat .ejs verwendet, welches eine einfach Übermittlung von Daten auf verschiedene Seiten ermöglicht. Der Aufbau von einer .ejs-Datei ist im Prinzip gleich wie eine normale HTML-Seite. Über den Syntax <% %> lassen sich die Backend Variablen im Frontend anzeigen.

Die render() Funktion sowie das .ejs Dateiformat beinhalten bereits einige Sicherheitsaspekte gegen XSS Attacken. Sie sorgen z.B dafür das Client Sonderzeichen wie z.B „</>“ als Strings ausgegeben werden und somit nicht vom Browser als Javascript Befehl erkannt werden. Für das erstellen von Notizen haben wir die render() Funktion verwendet, sodass ein Angriff hier schwierig wird.

Für die XSS-Attacke haben wir testweise die send() Funktion von express verwendet. Diese Funktion enthält keine Sicherheitsfaktoren gegen solche Attacken. Mit ihr gestaltet sich aber auch die Übertragung von Variablen schwieriger. Es wurde auch ein Session ID Cookie verwendet. Um an den Cookie im Browser auslesen zu können haben wir der send() Funktion die Parameter „<script>alert(document.cookie)</script>“ mitgegeben, sodass uns das Session Cookie auf einer neuen Seite als alert ausgegeben wird.

Zur Ermöglichung dieser Schwachstelle musste eine Einstellung der Cookies von Express verändert werden, damit alle Cookies ausgegeben werden. Ohne diese Veränderung wurde der Cookie "connect.sid". Dabei wurde der Wert der Variable "httpOnly" gleich "false" gesetzt.



Fehler in der Zugriffskontrolle (Broken Access Control)

|  |  |
| --- | --- |
| [1] | T. M. Heer, Lab Template, Esslingen: Heer's Pony Books, 2020. |