10. Januar 2018

Nadja Stadelmann, Tatyana Merlo, Natalie Stalder

TBZ

Analyse Schlosslauf

M183 – Applikationssicherheit implementieren

Inhaltsverzeichnis

[Einführung und Überblick 2](#_Toc503713708)

[Schlosslauf 2](#_Toc503713709)

[Softwareanalyse und vorhandene Schwachstellen 3](#_Toc503713710)

[Architektur 3](#_Toc503713711)

[Datalayer 3](#_Toc503713712)

[Businesslayer 3](#_Toc503713713)

[Presentationlayer 4](#_Toc503713714)

[Sicherheitsrisiken 4](#_Toc503713715)

[Weitere Mängel 5](#_Toc503713716)

[Bericht / Zusammenfassung 5](#_Toc503713717)

[Resultat nach Bereinigung der Lücken 6](#_Toc503713718)

[Architektur 6](#_Toc503713719)

[Datalayer 6](#_Toc503713720)

[Businesslayer 6](#_Toc503713721)

[Presentationlayer 6](#_Toc503713722)

[Sicherheitsrisiken 6](#_Toc503713723)

[Weitere Mängel 6](#_Toc503713724)

[Quellen 6](#_Toc503713725)

# Einführung und Überblick

evtl. noch manuell Angriffe durchführen

Security ist in der IT ein sehr wichtiges Thema. Gerade Webapplikationen, welche Personendaten beinhalten, sind ein beliebtes Ziel für Cyber-Attacken. Dies kommt daher, dass Daten immer wichtiger werden und mit ihnen auch einen grossen Schaden für Firmen und auch Einzelpersonen angerichtet werden können. Deshalb ist es umso wichtiger, seine Applikationen vor Angriffen zu schützen.

Im Rahme des Moduls 183 gehen wir genauer auf die Angriffsmöglichkeiten und Gefahren ein, damit wir fähig sind, unsere Webapplikation sicher zu halten. Um praktische Erfahrung im Lösen von Sicherheitsproblemen zu sammeln, analysieren und verbessern wir das Projekt Schlosslauf. Darin sind einige Mängel vorhanden, die Sicherheitsrisiken bilden. Diese gilt es zu erkennen, zu beschreiben und zu beheben. Weiter sind darin auch einige Logik- und Best-Practice-Fehler enthalten, die wir benennen werden. Der Fokus liegt jedoch nicht auf ihnen, weshalb wir nicht alle davon umsetzen werden.

## Schlosslauf

Die Webapplikation «Schlosslauf» soll dazu dienen, dass sich Leute an einem Lauf durch ein Schloss registrieren können. nach dem Login können sie ihre Daten anpassen, wenn gewünscht. Momentan kann nur jemand, der Zugriff zu einem bestehenden Konto hat, die Daten einfügen. Daher ist die Registrierung noch nicht unterstützt. Weiter soll die Applikation erweitert werden, sodass die User verschiedene Rollen haben, etwa einen Admin, der alles bearbeiten und ansehen kann, und ein Läufer, der nur die eigenen Daten begutachten kann.

Ergänzungen

# Softwareanalyse und vorhandene Schwachstellen

## Architektur

Einführung überarbeiten

Bei Betrachtung der Filestruktur fällt als erstes auf, dass bloss ein Ordner existiert. Dieser beinhaltet alle Files wild durcheinander. Das heisst, es besteht keine übersichtliche Trennung zwischen der Logik, den Daten und der Darstellung. Es existieren keine HTML-Dateien, das heisst, es besteht eine Vermischung zwischen den PHP- und den HTML-Dateien. Betreffend des Datenlayers ist ersichtlich, dass ein .txt-File und ein .sql-File existieren.

Bild einfügen

### Datalayer

Das angesprochene Textfile «file.txt» enthält diejenigen Daten, welche in dem Webformular eingegeben wurden. Diese werden für jeden Eintrag untereinander aufgelistet. Die Datenbank «schlosslauf.sql» besteht aus einer einzigen Tabelle, die Logindaten speichert.

Generell macht es keinen Sinn Daten dieses Formats in ein Textfile zu speichern, vor allem, wenn bereits eine Datenbank besteht. Wenn Daten aus dem File verändert werden sollten, ist dies eher umständlich zu lösen, während eine Datenbank bereits einfachere und sichere Mittel dafür integriert hat. Hier besteht also Handlungsbedarf.

In dem Script «schlosslauf.sql» sind Standardlogindaten gespeichert. Dies kann zu grossen Sicherheitsproblemen führen, sollte der Script in falsche Hände gelangen. Der Angreifer könnte sich mit den gefundenen Daten einloggen und alles ausführen, was der entsprechende User kann. Je nach dem, über welche Berechtigungen dieser verfügt, kann so ein grosser Schaden angerichtet werden.

#### Weiteres Vorgehen

Die Datenbank wird neu strukturiert und das Textfile entfernt. Dabei werden die Daten aus dem Textfile begutachtet und ein entsprechendes Datenmodell erstellt. Dadurch entsteht ein neues Script, in dem auf keinen Fall Standardlogindaten eingefügt werden. Diese Änderungen ziehen auch einige Logikänderungen mit sich, die gesamte Datenspeicherung muss vom Beschreiben eines Files in eine Datenbankanbindung umgestellt werden.

### Businesslayer

Die Ordnerstruktur selbst gibt keine Auskunft über den logischen Aufbau der Applikation. Um einen Überblick zu erhalten, haben wir daher das Tool «Rips» angewandt, welches unter anderem eine Graphik erstellt, aus der ersichtlich wird, welche Dateien wo eingebunden werden. Daraus ist ersichtlich, dass die Datei «index.php» der Eintrittspunkt der Applikation darstellt. Einen einzigen Eintrittspunkt (Single Point of Entry) zu haben ist von Vorteil, da so einige Sicherheitsprüfungen vereinfacht umgesetzt werden können. Dieser Eintrittspunkt sollte jedoch auch so umgesetzt werden, dass die anderen Files nicht einfach so aufgerufen werden können. Momentan kann ein nicht eingeloggter User alle Files öffnen oder herunterladen, wenn er den zugehörigen Namen in der URL einfügt. Dies muss abgefangen werden.

Rips Bild einfügen

Eine weitere Analyse der Indexdatei hat ergeben, dass für die Anzeige der Navigation und das Anzeigen der Inhaltsseiten überprüft wird, ob ein User eingeloggt ist. Dies ist grundsätzlich gut. Besser wäre es jedoch, zwei Templates zu erstellen, eines für die eingeloggte Ansicht und eines für den öffentlichen Bereich. So kann die doppelte Ausführung des Session-Checks umgangen werden, um Fehler zu vermeiden und Ressourcen zu schonen.

#### Weiteres Vorgehen

Die Ordnerstruktur wird überarbeitet um die Übersicht über die Webapplikation zu gewährleisten. Dies reduziert auch sicherheitsrelevante Fehler, die ansonsten aufgrund fehlender Übersicht nicht entdeckt werden würden.

Der Single Point of Entry wird durch Login-Überprüfungen in allen relevanten Files durchgesetzt. Ist kein User eingeloggt, so wird immer die Login- bzw. die Registrierungsseite des Schlosslaufs angezeigt. Dies reduziert das Risiko, dass ein Nicht-Autorisierter auf Inhalte zugreifen kann, die nicht für ihn bestimmt sind.

Um die zwei Templates umzusetzen, müsste die HTML-Struktur aus dem Indexfile ausgelagert werden. Da dies jedoch nicht sicherheitsrelevant und damit nicht erste Priorität dieser Arbeit ist, verzichten wir vorerst auf diese Verbesserung.

### Presentationlayer

In der bereits angesprochenen Datei «index.php» findet sich auch die Anzeigelogik der Website. Die darin eingebundenen «.inc.php»-Dateien stellen zusammen mit der im File integrierten HTML-Struktur die Webseite dar. Die meisten Dateien beinhalten reines HTML, Ausnahmen sind die Dateien navigation.inc.php und kopf.inc.php, welche Anzeige-Logik beinhalten. Die Vermischungen von verschiedenen Codearten sowie das unterschiedliche Handhaben von derselben Art Logik sollte vermieden werden. Die anderen «inc.php»-Dateien könnten direkt als HTML-Files gespeichert werden, dies würde dem PHP-Interpreter einiges an Arbeit ersparen. Weiter sollten die Standardtags sich nicht in dem Indexfile befinden, sondern in einem HTML-Template besser von der Logik abgetrennt werden.

Die Formulare für das Login und für die Schlosslaufanmeldung werden unterschiedlich gehandhabt. Die Anzeige des Loginformulars wird in einem eigenen File gespeichert, das Anmeldeformular ist mit der Logik im selben File eingebunden. Grundsätzlich sollten gleiche Dinge gleich gehandhabt werden, also sollte man sich für eine Bearbeitunsart entscheiden. Eine Trennung der Ansicht und der Logik ist für die Wartung von Vorteil.

Der Zweck der Webapplikation «Schlosslauf» (siehe Kapitel Einführung, Schlosslauf) sieht ein Signup vor. Diese wurde bisher jedoch noch nicht umgesetzt.

Für das Styling der Schlosslaufwebseite wird die CSS-Datei «screen.css» verwendet. Die zweite CSS-Datei «style.css» wird unseres Erachtens nirgends verwendet und kann daher entfernt werden.

überprüfen ob style.css wirklich nicht verwendet wird

#### Weiteres Vorgehen

Im Rahmen der Änderung der Ordnerstruktur werden auch die Dateien mit der Benennung «.inc.php» in einen eigenen Ordner verlagert. Deshalb können sie zur Vereinfachung und besseren Lesbarkeit umbenennt werden. Wie bereits im vorhergehenden Kapitel erwähnt verzichten wir im Rahmen dieses Moduls aus Zeitgründen auf die Trennung von HTML und PHP – und bilden somit kein HTML-Template. Um dennoch die Übersicht zu verstärken, werden die reinen HTML-Files neu auch als datei.html gespeichert. So ist schneller erkennbar, in welchen dieser Files sich Logik verbirgt.

Das Registrierungsformular für den Schlosslauf wird aus dem Registrationscontroller ausgelagert, um die Formulare einheitlich zu handeln.

Das Signup muss umgesetzt werden, damit der Zweck der Applikation erfüllt werden kann.

Die unnötige Datei «style.css» birgt zwar keine Sicherheitsrisiken, dennoch wird sie aus dem Projekt entfernt.

## Sicherheitsrisiken

## Weitere Mängel

## Bericht / Zusammenfassung

ist das überhaupt nötig?

🡪 hier klassifizierung vornehmen!!

- Datenbankanpassungen

# Resultat nach Bereinigung der Lücken

## Architektur

### Datalayer

Neues Design einbinden

Die Datenbank sieht nun etwas anders aus. Wie in dem ERD zu erkennen ist, wurde die anfänglich existierende Logintabelle zusammen mit den Daten aus dem Textfile zu einer Usertabelle umgewandelt. Nach der Normalisierung sind ausserdem einige weitere Tabellen hinzugekommen, wie zum Beispiel die Gruppentabelle. Die Attribute sind einfachheitshalber alle auf «NOT NULL» eingestellt. Dies müsste in einem späteren Schritt nochmals überarbeitet werden.

Um die Sicherheit des Logins zu erhöhen, speichern wir ausserdem einen Salt ab, mit welchem das Passwort vor dem Hashen verschmolzen wird. Ausserdem ist kein Standardlogin mehr in dem Script vorhanden, welches von einem Angreifer benutzt werden könnte.

Weiter speichern wir ausserdem ab, ob der User Administratorrechte besitzt oder nicht. Standardmässig wird der User ohne Administratorrechte erstellt. Dies ist für die Logik der gesamten Webseite nötig (siehe Kapitel Einführung, Schlosslauf). Zusätzlich ist dies eine Erhöhung der Sicherheit, da so nur wenige Benutzer alle Daten einsehen können. Das reduziert das Risiko eines Angriffes enorm, da so spezifische Useraccounts gehackt werden müssen, um die Kontrolle über die gesamte Applikation zu erhalten.

### Businesslayer

Die Ordnerstruktur wurde erneuert, sodass man sich einfacher in der gesamten Umgebung zurechtfindet. Daraus wird wertvolle Zeit gewonnen – statt einer unendlich langen Suche in allen Files kann mehr Zeit in das Bereinigen von Schwachstellen investiert werden.

Neue Struktur einbinden

Der Single Point of Entry ist so umgesetzt, dass alle Zugriffe von nicht-eingeloggten Personen automatisch auf die Loginseite verweisen (ausgenommen der Registraturseite, diese muss natürlich ersichtlich bleiben).

### Presentationlayer

Im neuen Ordner «view» sind alle ehemaligen «.inc.php»-Dateien vorhanden. Nun sind die Webseiteninhalte viel schneller zu finden und somit einfacher wartbar als zuvor. Kostbare Zeit kann gespart werden. Sofort ist erkennbar, welche Dateien Anzeigelogik enthalten und welche nicht (HTML-Dateien).

Nochmals überprüfen und neue Files .html benennen, wenn nötig

Alle Formulare sind nun in eigenen Dateien zu finden, so auch das Anmeldeformular und das neue Signup-Formular. Die Logik dahinter ist im Ordner «controller» zu finden. Das Signup wurde so umgesetzt, wie es der Applikationszweck vorsieht.

## Sicherheitsrisiken

## Weitere Mängel

# Quellen

|  |  |
| --- | --- |
| Name | Link |
| Rips | https://sourceforge.net/projects/rips-scanner/?source=typ\_redirect |
| PHP Risiken | https://www.owasp.org/index.php/PHP\_Security\_Cheat\_Sheet |
|  |  |
|  |  |
|  |  |