LU02f - LB2 - Projekt-Themen: SQLi

1. Einleitung

Dieses Thema ist gerade deswegen spannend, weil in dieser alle drei Schichten einer 3-Schichten-Architektur miteinbezogen werden müssen.

1. Schicht: Sie benötigen ein Frontend (Selbst programmieren oder Postman verwenden)
2. Schicht: Sie müssen einen Node-Server programmieren.
3. Schicht: Es braucht eine sinnvolle Datengrundlage, um SQL-Attacken durchführen zu können.

2. Informationsquellen

Es gibt jede Menge Tutorials, sodass es wenig Sinn macht weitere zu erstellen. Sprich es ist Ihre Aufgabe sich entsprechend zu informieren. Nachfolgend eine kleine Auswahl von Quellen mit Substanz.

- SQL Injection: Wie es funktioniert und wie man es verhindern kann
- Wikipedia-SQLi
- SQLi-Emulator
- Sinnvolle Info mit Codebeispielen
- W3School-SQLi
- Weitere Link bitte mir melden, dann nehmen ich es hier in die List auf

3. Gegenmassnahmen gegen SQLi

3.1 Leichte Massnahmen

- Escaping: Spezielle Zeichen in Eingaben werden maskiert, damit sie von der Datenbank als Daten und nicht als Teil des SQL-Codes interpretiert werden.
- Input Validation: Überprüfung, ob Eingaben dem erwarteten Format, Typ oder Wertebereich entsprechen, bevor sie weiterverarbeitet werden.

3.2 Mittel-Komplexe Massnahmen

- Prepared Statement: die technische Umsetzung auf DB-Seite (vorkompiliertes Statement, wiederverwendbar).
- Parametrisierung: das Konzept, Werte sauber von der SQL-Struktur zu trennen.

3.3 Komplexe Massnahmen

- Stored Procedure: Vorgefertigte und in der Datenbank gespeicherte SQL-Routinen, die mit Parametern aufgerufen werden können, statt dynamisch SQL zusammenzubauen.

4. Bewertung

Nachfolgend finden Notenstufen bzw. die Anforderungen, um diese Noten zu erreichen.

Note 4.0 - 4.5

Konzept

- Es sind 4-5 UseCases formal korrekt und vollständig vorhanden
- ERD ist korrekt vorhanden mit mindestens 3 Tabellen (mindestens 1 Relation)
- Software-Architekturmodell (UML, …) vollständig und fachlich korrekt vorhanden

Datengrundlage

- Das konzipierte Datenbankmodell wurde korrekt in einer Datenbank umgesetzt.
- Die Datenbanktabellen enthalten eine represantive Datenmenge (>= 10 Datensätzen) in jeder Tabe

Software/Programmierung

- Es ist eine vulnerabler (node.js) Server mit einer Datenbankabindung vorhanden
- Es können Angriffe gegen eine der CIA-Triads erfolgreich durchgeführt werden: Vertraulichkeit
- Es gibt eine *secure-Version* der Software, bei der erfolge Angriff mittels eine der unter 3.1 genannten Massnahmen verunmöglicht wurde.

Formale Aspekte

- Die Best-Practice-Coding-Standard wurden eingehalten
- Es wird colaborativ gearbeitet inkl. einer Ausfallsicherheit und Versionierung

Abgrenzung

- Es braucht kein Frontend, sprich der Code kann entweder über Commandline oder als separates File eingelesen werden.
- Sprich der Fokus liegt auf dem Server

Note 4.5 - 5.2

Konzept

- Es sind 6-8 UseCases formal korrekt und vollständig vorhanden
- ERD ist korrekt vorhanden mit mindestens 4 Tabellen (mindestens 1 Relation)
- Software-Architekturmodell (UML, …) vollständig und fachlich korrekt vorhanden

Datengrundlage

- Das konzipierte Datenbankmodell wurde korrekt in einer Datenbank umgesetzt.
- Die Datenbanktabellen enthalten eine represantive Datenmenge (>= 20 Datensätzen) in jeder Tabe

Software/Programmierung

- Es ist eine vulnerabler (node.js) Server mit einer Datenbankabindung vorhanden
- Es können Angriffe gegen eine der CIA-Triads erfolgreich durchgeführt werden: Vertraulichkeit und Integrität
- Es gibt eine *secure-Version* der Software, bei der erfolge Angriff mittels je eine der unter 3.1 und 3.2 genannten Massnahmen verunmöglicht wurde.

Formale Aspekte

- Die Best-Practice-Coding-Standard wurden eingehalten
- Es wird colaborativ gearbeitet inkl. einer Ausfallsicherheit und Versionierung

Abgrenzung

- Als Ersatz für das Frontend kann Postman oder curl verwendet werden
- Sprich der Fokus liegt auf dem Server und eine Kommunikation mit einem potentiellen Client (Ersatz)

Note 5.3 - 6.0

Konzept

- Es sind >9-12 UseCases formal korrekt und vollständig vorhanden
- ERD ist korrekt vorhanden mit mindestens 5 Tabellen (mindestens 2 Relation)
- Software-Architekturmodell (UML, ...) vollständig und fachlich korrekt vorhanden

Datengrundlage

- Das konzipierte Datenbankmodell wurde korrekt in einer Datenbank umgesetzt.
- Die Datenbanktabellen enthalten eine represantive Datenmenge (>= 30 Datensätzen) in jeder Tabe

Software/Programmierung

- Es ist eine vulnerabler (node.js) Server mit einer Datenbankabindung vorhanden
- Es können Angriffe gegen eine der CIA-Triads erfolgreich durchgeführt werden: Vertraulichkeit und Integrität
- Es gibt eine *secure-Version* der Software, bei der erfolge Angriff mittels je eine der unter 3.1, 3.2 und 3.3 genannten Massnahmen verunmöglicht wurde.

Formale Aspekte

- Die Best-Practice-Coding-Standard wurden eingehalten
- Es wird colaborativ gearbeitet inkl. einer Ausfallsicherheit und Versionierung

Abgrenzung

- Als Ersatz für das Frontend kann Postman oder curl verwendet werden
- Sprich der Fokus liegt auf dem Server und eine Kommunikation mit einem potentiellen Client (Ersatu)