LU02e - LB2 - Ihalt: 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 eine Art Frontend-Simulation wie beispielsweise Postman oder Commandline-basiertes Backend-Script. Es braucht also keine schicke Oberfläche.
2. Schicht: Sie müssen einen Node-Server programmieren.
3. Schicht: Es braucht eine sinnvolle Datengrundlage, um SQL-Attacken durchführen zu können.

2. SQLI-Angriffe

2.1 Leicht umsetzbare Angriffe

- **Angriffe gegen die Vertraulichkeit:** Es können Daten aus einer Tabelle gelesen werden, die nicht gelesen werden dürften.

2.2 Mittel umsetzbare Angriffe

- **Angriffe gegen die Verfügbarkeit:** Das System wird zum Absturz gebraucht, sodass Online-Dienste nicht mehr angeboten werden können.

2.3 Schwer umsetzbare Angriffe

- **Angriffe gegen die Integrität:** Die Daten werden verändert, ohne dass die betroffene Person davon weiss.

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

4.1 Angriff

Noten	Beschreibung	Hinweis
4.1.1	+0.75	Angriff gemäss einer Variante von 2.1
4.1.2	+0.75	Angriff gemäss je einer Variante und 2.2
4.1.3	+1.00	Angriff gemäss je einer Variante 2.3
4.1.4	+0.50	Angriff ist jeweils in separaten Files untergebracht

4.2 Verteidigung

Noten	Beschreibung	Hinweis
4.2.1	+0.75	Verteidigung gemäss einer Variante von 3.1
4.2.2	+0.75	Verteidigung gemäss je einer Variante und 3.2
4.2.3	+1.00	Verteidigung gemäss je einer Variante 3.3
4.2.4	+0.50	Verteidigung sind in jeweils separaten Files untergebracht

4.3 Bonus

4.3.1	+0.25	Die Thementeams werden in Konkurenz zueinander treten. Sofern ein wesentzlicher Teambeitrag vorhanden ist, erhält jedes Gewinnerteam jeweils einen Bonus von 0.25 Notenpunkten.
4.3.2	+0.50	Für die 2. und 3. Angriffe oder Verteidigungen wird die Hälfte der vollen Punktezahl der jeweiligen Massnahme vergeben
4.3.3	+0.50	SQLi Daten sind in einer XML oder JSON-Datei

4.4 Malus

Noten	Beschreibung	Hinweis	
4.4.1	-1.00	Die Best-Practice-Coding-Standard wurden nicht eingehalten	Python
4.4.2	-0.50	Es wird nicht kollaborativ und Ausfallsicher (inkl. Versionierung) gearbeitet	
4.4.3	-2.00	Es können Fragen zum Lösungcode nicht korrekt oder gar nicht beantwortet werden.	
4.4.4	-1.50	Pro angefangenem Tag Verspätung	
4.4.5	-0.50	Pro Downloadversuch: Link funktioniert nicht (abgelaufen, Berechtigungen fehlen, etc.)	
4.4.6	-0.50	Schriftsprache: Die Lernenden verwenden nicht die offiziellen Schriftsprache.	

Noten	Beschreibung	Hinweis	
4.4.7	-0.10	Verspäteter Anfang des Teams: pro Minute Verspätung	
4.4.8	-1.00	Team oder Teammitglieder erscheinen nicht zum Gespräch (Ausnahme: Höhere Gewalt). Abzug erhalten nur betroffenen Personen	
4.4.9	-1.00	Team ist nicht vorbereitet beim Fachgespräch	

4.5 Hinweise

- Es braucht kein Frontend, sprich der Code kann entweder über Commandline oder als separates Angriffsscriptdurchgeführt werden.
- Jedes Teammate kann alle Fragen zum Fousthema beantworten, sprich der Wissentransfer im Tandem wird durch das Tandem sichergestellt.
- Bei Ungleicher Performance im Tandem wird individuell benotet.
