SQL Injection Primer

Raphael Karger & Mike Antoniades

Housekeeping

- Attendance - https://forms.gle/f4FSQEGQ6TE51aGQ7

Structured Query Language (SQL)

- Used for managing data held in relational database management system
- Industry standard
- Capabilities include:
  - Executing queries
  - Retrieving data
  - Inserting records
  - Deleting records

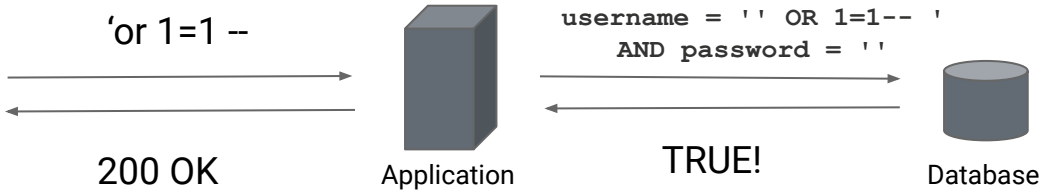
SQL Injection

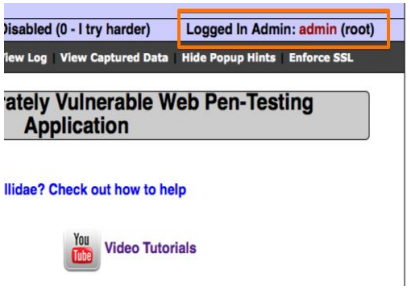
- SQLi vulnerabilities arise when user supplied data becomes part of SQL queries in an unsafe manner.
- An attacker can inject a malicious input and execute SQL commands leading to reading and/or modifying the stored data and sometimes even performing remote code execution.

Login Bypass Example

- Eg. SELECT * FROM users WHERE user =’<user input>’ AND pass=’<userinput>’
- Payload: ‘or 1=1 --
- ---> SELECT * FROM users WHERE user=’’or 1=1 --’ AND password=’foo’
    - Query Evaluates to true - Login Bypassed

Login Bypass Example





Boolean Based

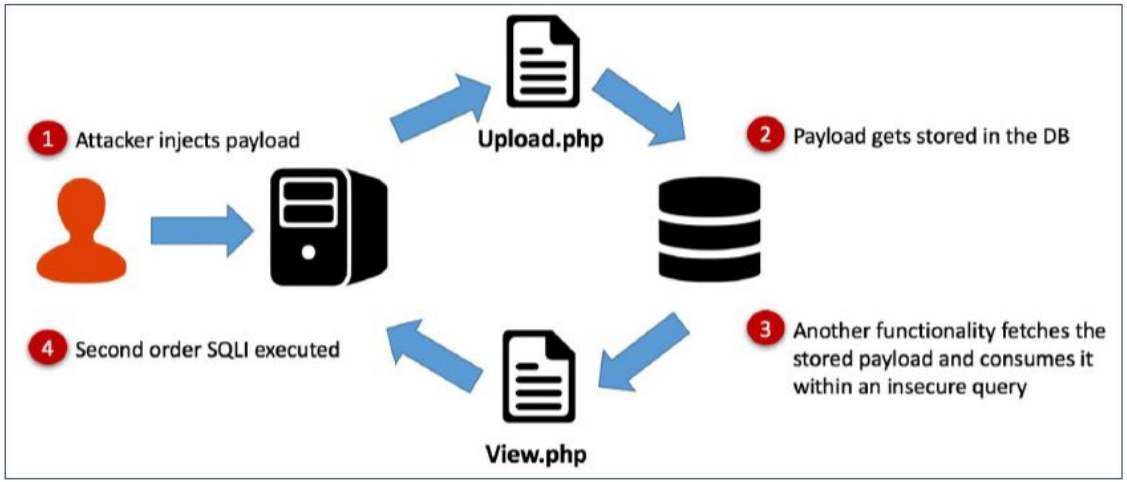
- Forcing query to return either a true or false response
- Allows an attacker to determine whether a query was true or false without having access to errors
- Very slow as an attacker would need to enumerate the data base one character at a time

Blind (Inferential) SQL Injection

- User input is improperly put into a query however no output can be seen
- Determining the value of a query is usually determined in a boolean value by utilizing time
- Tends to be very slow, as each character must be enumerated in database
- Depending on the timings false positives could be common

Second Order Injection

- Leveraging file upload/creation functionalities to upload/create .php file with malicious SQL queries
- Leveraging website functionality to load the .php file
- Custom SQL Query is executed



Activity

- Vulnerable application - https://b7a73ec04fb3.ngrok.io/
- SQLi vulnerabilities are present
- Exploit them!