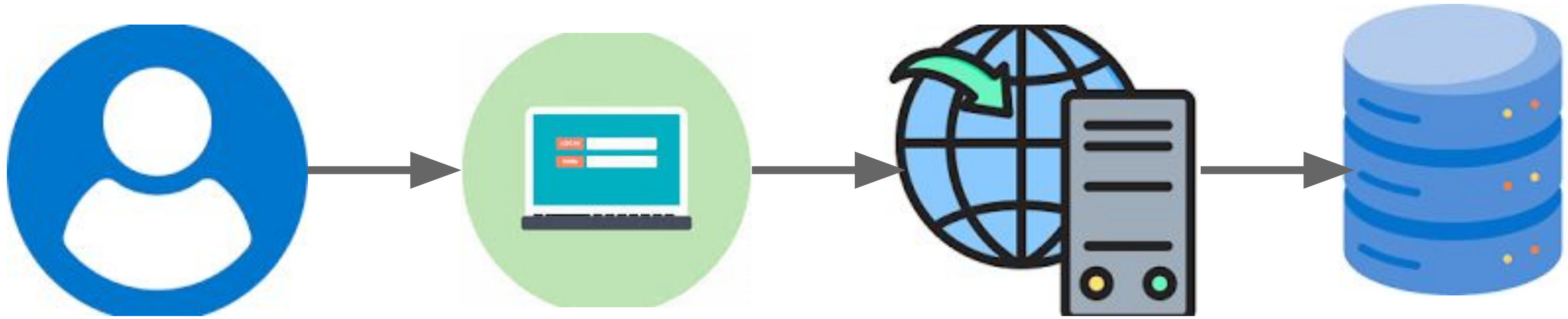
Web Security

Accessing Web Services



Relational Databases (RDBMS)

- The data in a relational database is stored in the form of tables. A table is a collection of related data entries and it consists of columns and rows.
- Database are accessed/edited using SQL statements.

Employee

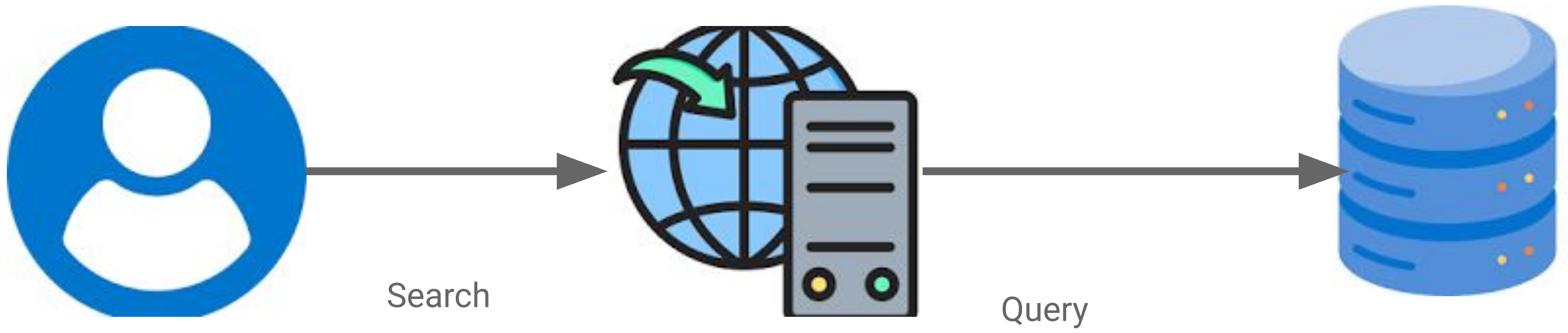
ID	Name	Designation	Location
1	A	Manager	Mumbai
2	B	Team Lead	Bengaluru
3	C	Software Engineer	Bengaluru

SQL Commands

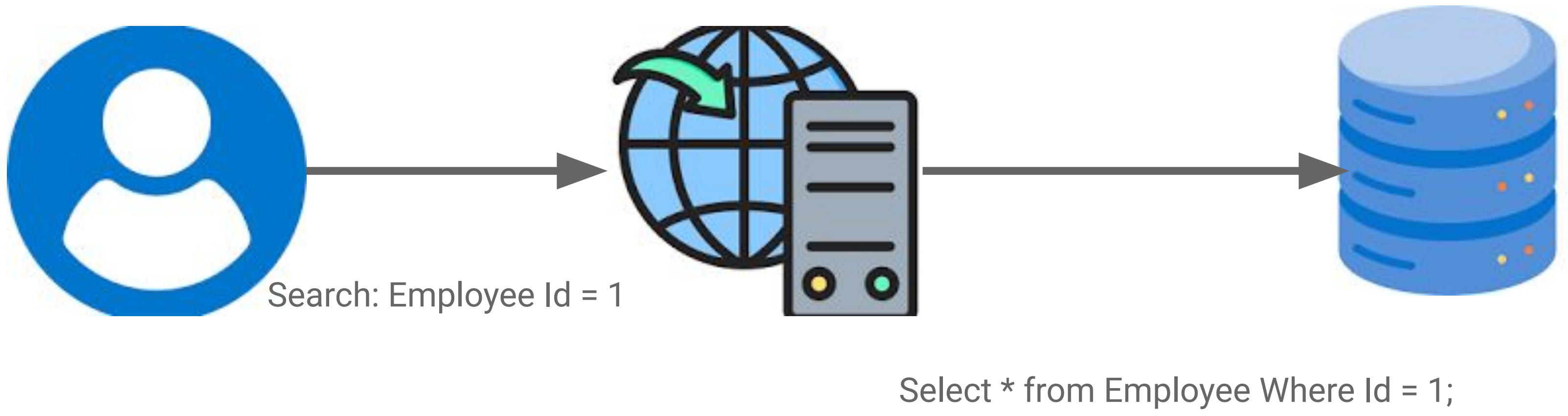
- CREATE DATABASE - creates a new database
- CREATE TABLE - creates a new table
- ALTER TABLE - modifies a table
- DROP TABLE - deletes a table
- SELECT - extracts data from a database
- UPDATE - updates data in a database
- DELETE - deletes data from a database
- INSERT INTO - inserts new data into a database

NOTE: Although SQL is an ANSI/ISO standard, there are different versions of the SQL language.

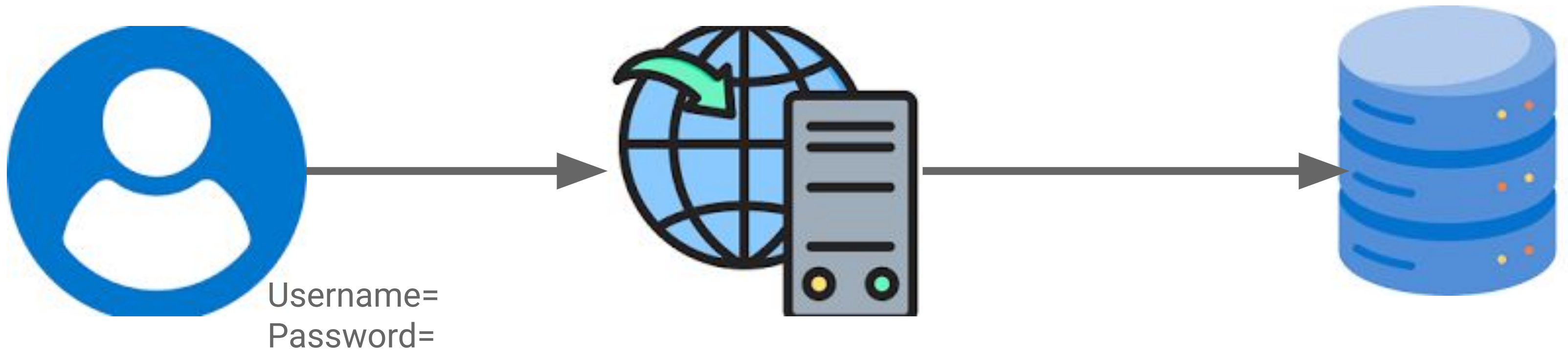
Accessing Web Services



Accessing Web Services

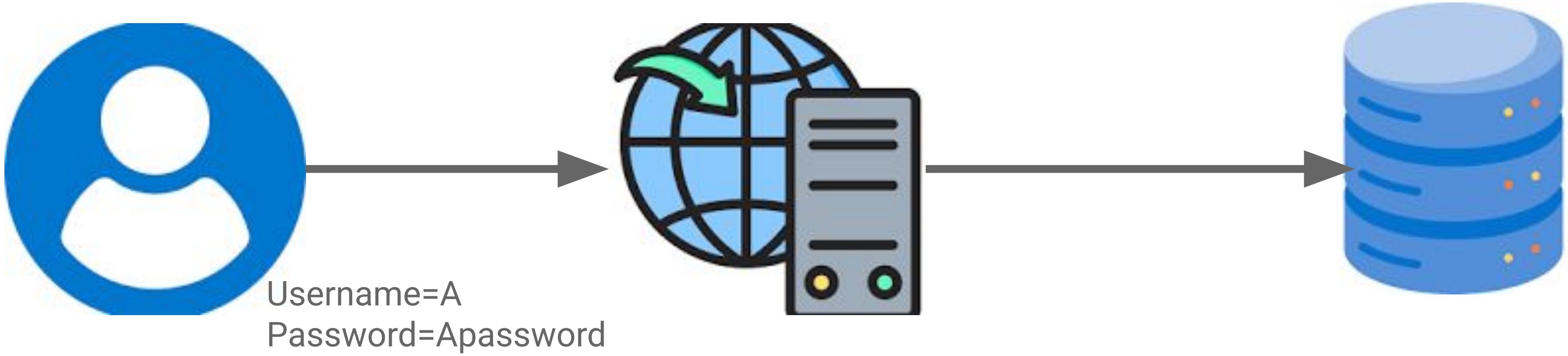


Accessing Web Services



Select * from Employee Where username = ‘ ’
AND Password = ‘ ‘;

Accessing Web Services

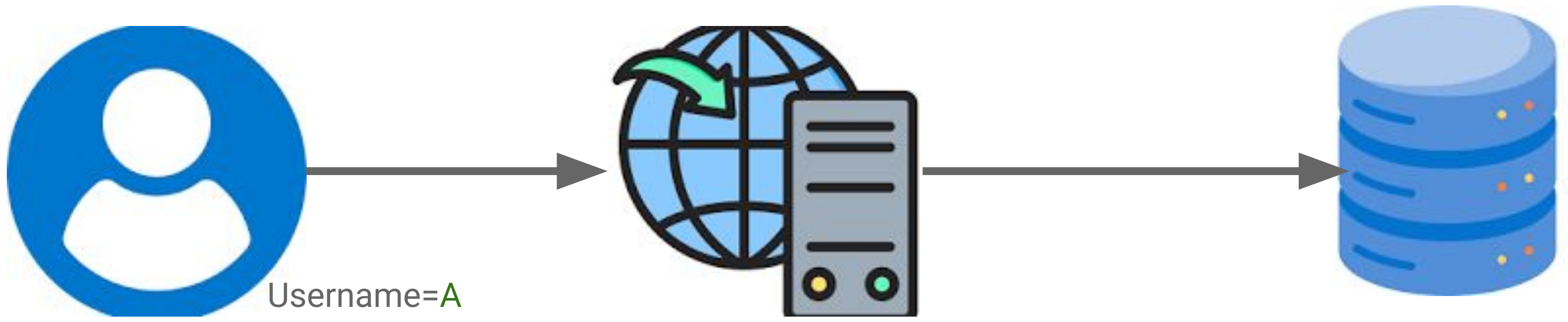


Select * from Employee Where username = 'A' AND Password = 'Apassword' ;

SQL Injection

- SQL Injection (SQLi) is a type of an injection attack that makes it possible to execute malicious SQL statements.
- Here SQL commands are injected into malicious input in order to affect the execution of predefined SQL commands.
- It may affect any website or web application that uses an SQL database such as MySQL, Oracle, SQL Server, or others.
- It is among the OWASP top 10 vulnerabilities
- This attack allows attackers to spoof identity, upload large files, tamper with existing data, cause repudiation issues such as voiding transactions or changing balances, allow the complete disclosure of all data on the system, destroy the data or make it otherwise unavailable, and become administrators of the database server.

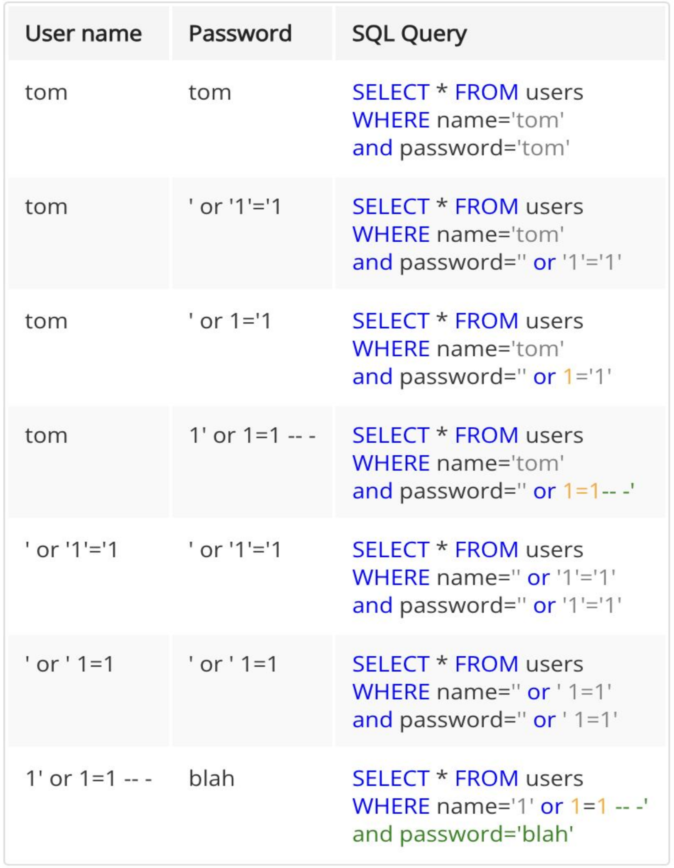
Accessing Web Services



Username=A
Password=’ OR ‘1’=’1

Select * from Employee Where username = ‘A’ AND Password = ‘ ’ **OR ‘1’=’1’** ;

SQL Injection Vectors



User name	Password	SQL Query
tom	tom	SELECT * FROM users WHERE name='tom' and password='tom'
tom	' or '1'='1	SELECT * FROM users WHERE name='tom' and password='' or '1'='1'
tom	' or 1='1	SELECT * FROM users WHERE name='tom' and password='' or 1='1'
tom	1' or 1=1 -- -	SELECT * FROM users WHERE name='tom' and password='' or 1=1-- -'
' or '1'='1	' or '1'='1	SELECT * FROM users WHERE name='' or '1'='1' and password='' or '1'='1'
' or ' 1=1	' or ' 1=1	SELECT * FROM users WHERE name='' or ' 1=1' and password='' or ' 1=1'
1' or 1=1 -- -	blah	SELECT * FROM users WHERE name='1' or 1=1 -- -' and password='blah'

Types of SQL Injection

1. **In-Band SQLi:** same communication channel is used for both launching the attack and gathering results.
   a. **Error-based:** Relies on error messages thrown by the database server to obtain information about the structure of the database
   b. **Union-based:** leverages the UNION SQL operator to combine the results of two or more SELECT statements into a single result which is then returned as part of the HTTP response.
2. **Blind SQLi:** the attacker would not be able to see the result of an attack in-band. Instead, an attacker is able to reconstruct the database structure by sending payloads, observing the web application's response and the resulting behavior of the database server.
   a. **Boolean-based:** relies on sending an SQL query to which forces the application to return a different result depending on whether the query returns a TRUE or FALSE result.
   b. **Time-based:** relies on sending an SQL query to the database which forces the database to wait for a specified amount of time (in seconds) before responding. The response time will indicate to the attacker whether the result of the query is TRUE or FALSE.
3. **Out-of-band SQLi:** relies on the database server's ability to make DNS or HTTP requests to deliver data to an attacker.

Impact of SQL Injection

- Unauthorized Access
- Data Manipulation
- Data Loss

Avoiding SQL Injection

SQL Injection happens when query and user data are combined

- Input validation
  - Whitelisting: Defining a list of values that are acceptable as user input.
  - Typecasting: If the user input has data types such as integers, boolean, then the user input can be typecasted before combining with queries
  - Escaping: Escaping special characters such as single and double quotes
- Prepared Statements/Parameterized queries: Queries are compiled before concatenating the user inputs
  - Query statements are prepared without user input, parsed, compiled and then user inputs are provided.
  - SELECT * FROM users WHERE name=? AND password= ?

Cross-Site Scripting (XSS)

- Code injection attack where attacker injects malicious code through web browser
- Here, when a victim clicks on a link or visits a website, the malicious code will execute on the browser
- Three main types of XSS Attacks include
  - Reflected XSS
  - Stored XSS
  - DOM-based XSS

XSS

Reflected

- Its non-persistent XSS i.e the malicious code/script is not stored on the server
- The script will be reflected on the victim’s browser

Stored

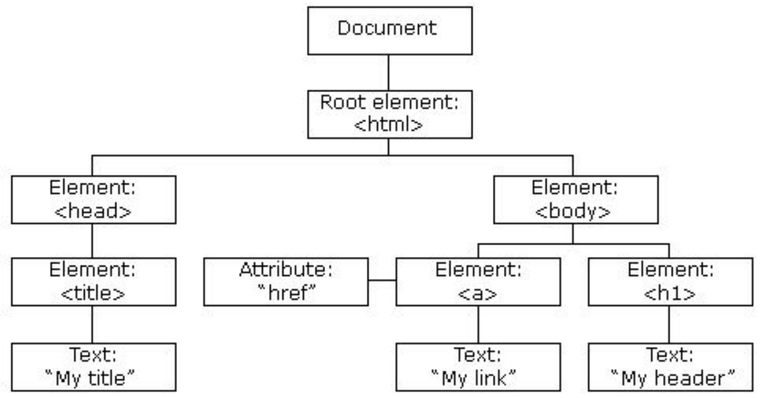
- Script is stored on the server
- It will be executed every time a user visits the webpage

DOM-based

- Attack payload is executed as a result of modifying the DOM “environment” in the victim’s browser

Document Object Model

- When a web page is loaded, the browser creates a Document Object Model of the page where it constructs a tree of objects in the web page.
- With the object model, JavaScript gets all the power it needs to create dynamic HTML



Impact of XSS

- Can read user information
- Can Hijack Sessions
- Can steal credentials
- Can install drive-by downloaded malwares

Cross-Site Request Forgery (CSRF)

- Here attacker tricks a victim into performing actions that they didn’t intend to
- It is a one-way attack i.e, the attacker cannot see the response from the server
- It can be prevented with the help of
    - CSRF Tokens
    - Same-site Cookies
    - Referer header

References

1. https://xss-game.appspot.com/
2. http://testphp.vulnweb.com/login.php
3. https://www.w3schools.com/sql/sql_injection.asp
4. https://portswigger.net/web-security/sql-injection#what-is-sql-injection-sqli
5. https://owasp.org/www-community/attacks/SQL_Injection
6. https://www.acunetix.com/websitesecurity/sql-injection/
7. https://owasp.org/www-community/attacks/xss/
8. https://www.acunetix.com/websitesecurity/cross-site-scripting/
9. https://www.w3schools.com/Jsref/event_onerror.asp