FINAL PROJECT

SAN JOSE STATE UNIVERSITY

CS166 SPRING 2017

KAYA OTA

CONTENT

- Behind the scene tour of this site.
- SQL Injection
- XSS (Cross Site Scripting)
- Cookie Stealing
- Protocol
  - Authentication
- Trojan House

- Behind the seen
- SQL Injection
- XSS (Cross Site Scripting)
- Cookie Stealing

- Authentication
- Trojan House

BEHIND THE SCENE TOUR OF THIS SITE

ENTRY URL FOR CS166 BLOG

- **Prevented codes are running at:**
- **http://ec2-34-208-99-244.us-west-2.compute.amazonaws.com:8080/CS166_Final_Project/Project_Code/prevented/index.html**
- **The given codes are running at:**
- **http://ec2-34-208-99-244.us-west-2.compute.amazonaws.com:8080/CS166_Final_Project/Project_Code/attackable/index.html**

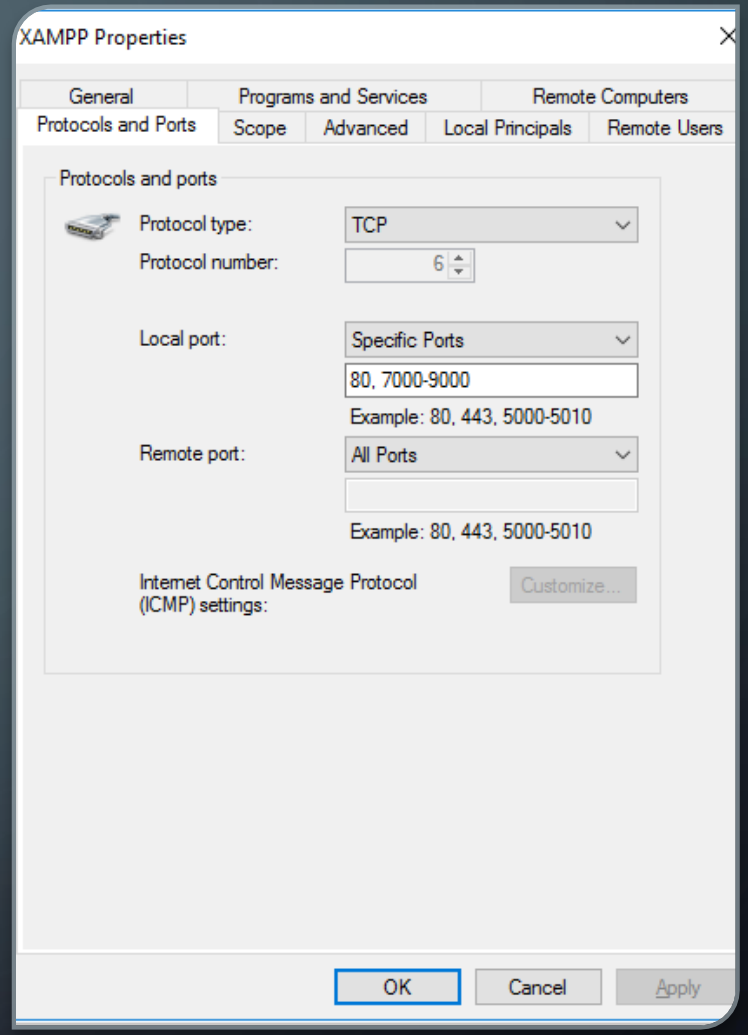
HOW TO BUILD THIS SITE

- Download source code from the git hub: https://github.com/28kayak/CS166_Final_Project.git
- Set up AWS windows server with the following security group.

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ
HTTP	TCP	80	0.0.0.0/0
HTTP	TCP	80	::/0
Custom TCP Rule	TCP	7000 - 9000	0.0.0.0/0
RDP	TCP	3389	0.0.0.0/0

HOW TO BUILD THIS SITE

- Set up XAMPP with Tomcat and Maria DB.
- Check Windows server side of fire wall's setting. (image on the left)
- Tomcat entry is on port 8080.



SQL TABLE – LOGIN –

- Use Maria DB
  - **Login** table contains user information

Fullname	User	pass	role	Random

  - Fullname – user's name
  - User – user ID
  - Pass – password
  - Random – salt for the password

SQL TABLE –BLOG–

- Blog table contains posts for the blog.

title	content	id

- Title is title of the post
- Content is the articles in the post
- ID is the post id and is the primary key

SQL INJECTION

SQL INJECTION – OVERVIEW –

- A type of injection attack
- A SQL injection attack is by “injection” of SQL query via input data from the client to the application.
- When SQL succeed the followings could happen
  - Read sensitive data
  - Modify DB data
  - Run administrative operation

SQL INJECTION – THREAD MODELING –

- SQL Injection lets attackers to spoof identity, and temper data in database.
- SQL Injection lets cause repudiation issues
  - Voiding transaction
  - Changing balance
- SQL injection is common with PHP and ASP
  - Because these older functional interfaces are widely used.
  - Nature of programmatic interface available
- J2EE and ASP.NET application are less likely to have easily exploited SQL injection.

SQL INJECTION – PREVENTION –

I. Use prepared statement / parameterized queries
   I. Prepared statement force the developers to first define all SQL code and then pass the required parameters later to the query.
   II. This allows DB to distinguish between code and data, independent from user-input.

SQL INJECTION – PREVENTION –

No Use of Prepared Statement

```
String user = request.getParameter( "user" );
String pass = request.getParameter( "pass" );
String sqlStr = "SELECT fullname FROM login WHERE user='" + user + "' and pass = sha2('"+ pass + "', 256)";
```

Use of Prepared Statement

```
String sqlStr = "SELECT count(*) FROM login WHERE user=? and pass = sha2(?, 256)";
PreparedStatement stmt = con.prepareStatement(sqlStr);
stmt.setString(1,name);
stmt.setString(2,pwd);
ResultSet rs = stmt.executeQuery();
```

SQL INJECTION – PREVENTION –

II. Use Stored Procedure
    I. Not always safe from SQL Injection
    II. Certain Stored Procedures have the similar effect as use of parameterized query
    III. It requires to build SQL query with parameters that are automatically parametrized unless the developer does something out of norm.

SQL INJECTION – DEMONSTRATION –

- Not Preventing Site
  - http://ec2-34-208-99-244.us-west-2.compute.amazonaws.com:8080/CS166_Final_Project/Project_Code/attackable/login_form.html
- Preventing Site
  - Running here

XSS – CROSS SITE SCRIPTING –

XSS – OVERVIEW –

- A type of injection attack
- Injects malicious script into benign and trusted website.
- Occurs when an attacker users a web application to send malicious code
- Generally in the form of a browser side script to different end user.

XSS – THREAD MODELING –

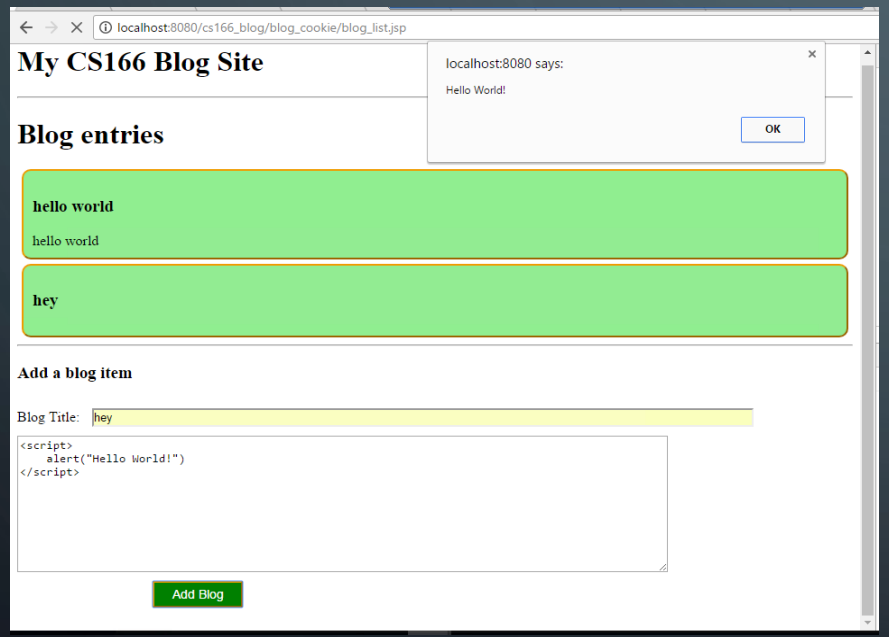
- XSS lets attackers do the followings
  - Identity Thrift (fraud)
  - Redirect traffic by altering URL
  - Session Hijacking
  - Storing sensitive information in JavaScript variables

XSS – PREVENTION –

- Never accepts to insert untrusted data except in allowed location
  - Deny all – do not put untrusted data into your html document unless it is within one of the slot of defined in rule #1
  - Most importantly, never accept actual JavaScript code from an untrusted data and then run it.
- Escape XML sequences
  - Using Escape sequences
  - http://www.avajava.com/tutorials/lessons/how-do-i-escape-a-string-for-xml.html

SCREEN SHOT FOR XSS

ATTACKED



PREVENTED

XSS –DEMONSTRATION–

- Demonstration running at
  - http://ec2-34-208-99-244.us-west-2.compute.amazonaws.com:8080/CS166_Final_Project/Project_Code/attackable/login_form.html

PROTOCOL

PROTOCL

- 5 REQUEIREMENTS

REPLAY ATTACK

REFERENCE

- https://www.owasp.org/index.php/SQL_Injection