An Introduction to Web Security

Ansh Bhawnani

Web Development

Principles

Colors

Purpose

Images

Simplicity

Performance

Security

But why Security?

- There is a hack attack every 39 seconds
- The average cost of a data breach in 2020 will exceed $150 million
- $2 trillion total in 2019
- $6 trillion is expected to be spent globally on cybersecurity by 2021
- Cybersecurity jobs worldwide will reach 3.5 million by 2021

"Cybercrime is the greatest threat to every company in the world."

How to define security?

“Human stupidity is the biggest vulnerability.”

Website Security Principles

01 Confidentiality

02 Integrity

03 Availability

04 Authentication

What should you know, as a Developer?

Top 8 web security threats

1. Injection

Injection

01

1. SQL Injection

02

2. Command Injection

03

3.. XML Injection

04

4.. LDAP Injection

1. SQL Injection

Normal input: http://www.example.com/?user_id=1

Malicious input: http://www.example.com/?user_id=1**’ or 1=‘1**

Backend Query:
‘select * from users where user_id=1’

Manipulated Query:
‘select * from users where user_id=1’ or 1=‘1’ (always true)

2. Command Injection

Normal input: http://www.example.com/?ip_addr=134.32.34.11

Malicious input: http://www.example.com/?ip_addr=134.32.34.11&dir

Backend Query:

ping 134.32.34.11

Manipulated Query:

ping 134.32.34.11**&dir**

“A user input is always malicious”

2. Broken Authentication

Broken Authentication

1. URL Exposure:http://www.example.com/user/101/?session=234739jf7932th0f
2. Weak Session IDs
3. Set-Cookie: eyt3nfo (fixation)
4. Session Hijacking

http://website.kom/<script>document.cookie=”sessionid=abcd”;</script>

Broken Authentication

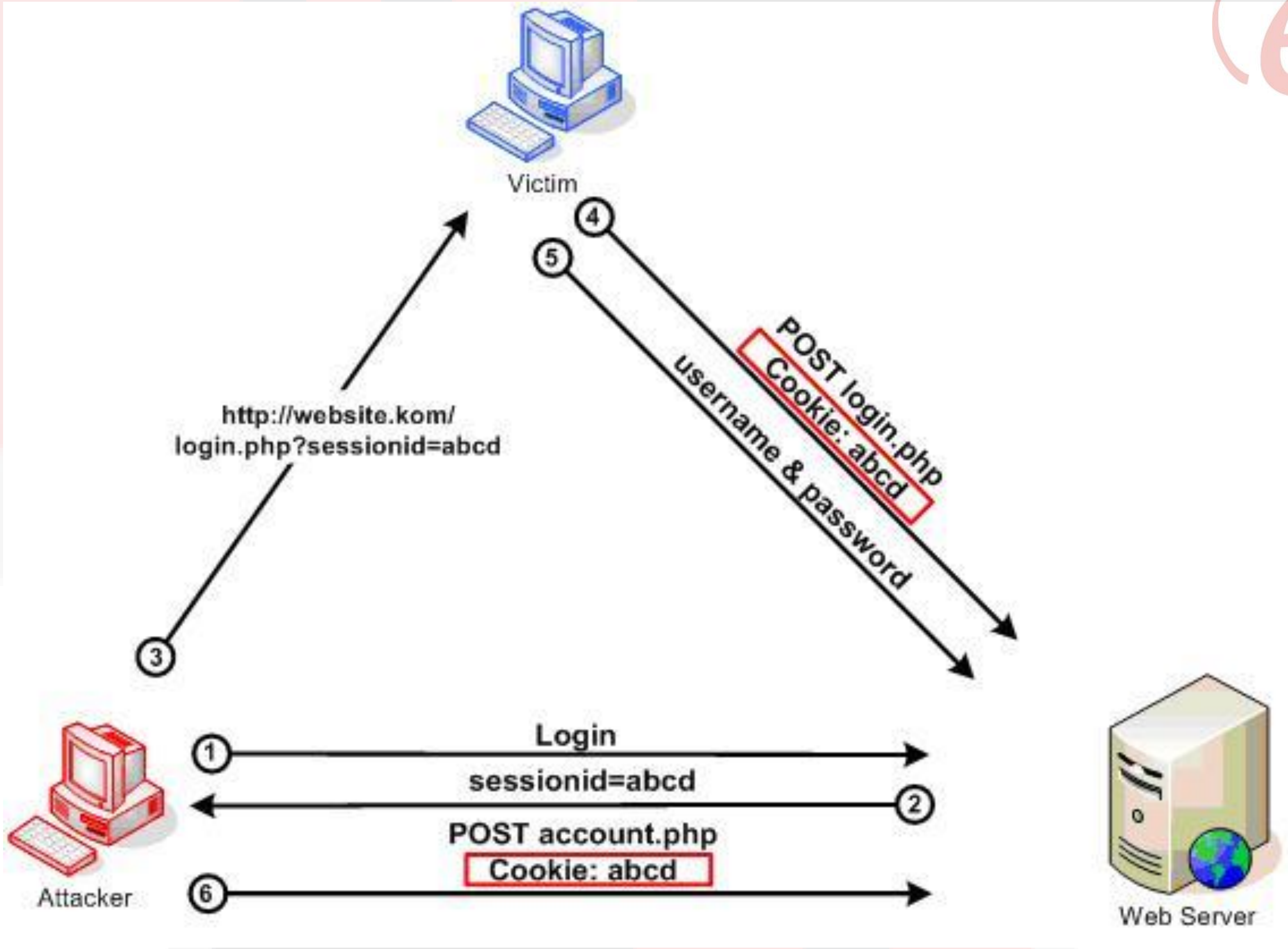
5. Weak Remember Password

   Cookie: session=234nf83h34t; password=something

6. Successive login attempts

7. Email verification

8. Weak Username/Password policies (charset, length)


Victim
Attacker
Web Server
1 Login
2 sessionid=abcd
3 http://website.kom/ login.php?sessionid=abcd
4 POST login.php Cookie: abcd
5 username & password
6 POST account.php Cookie: abcd

“Your Identity is everything.”

3. Sensitive Data Exposure

Sensitive Data Exposure

1. HTTPS

2. HTTP Strict Transport Policy (hack-yourself-first.com)

http://google.com
https://google.com

Sensitive Data Exposure

1. In transit:
   1. Plain text

2. Stored data exposure
   1. Storing passwords in plain text
   2. No hashing
   3. No salting

3. Cookies Secure flag

“Don’t tell everything you know.”

4. Cross Site Scripting (XSS)

Cross Site Scripting

1. Reflected

2. Stored

3.. DOM Based

Cross Site Scripting

1. Injecting arbitrary JavaScript codes in input strings

Normal URL: http://somesite.com/?user=John

HTML:

```
<body>
  <h1>Welcome John!</h1>
</body>
```

Cross Site Scripting

1. Injecting arbitrary JavaScript codes in input strings

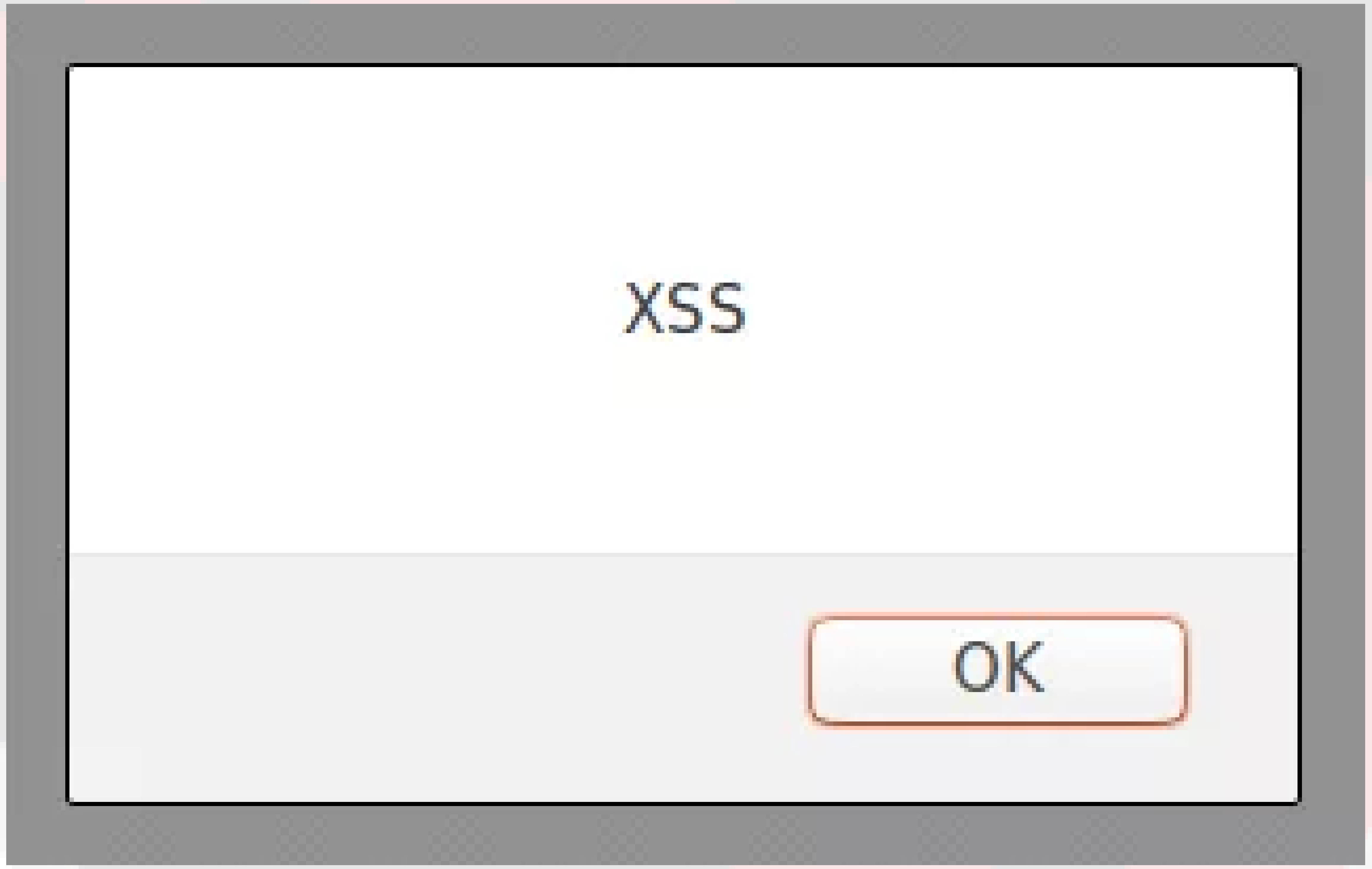
Malicious URL: http://somesite.com/?user=**<script>alert('XSS')</script>**

HTML:

```
<body>
  <h1>Welcome <script>alert('XSS')</script>!</h1>
</body>
```

Cross Site Scripting

Result:



XSS

Cross Site Scripting

2. Permanently store malicious JS code in DB

Malicious Request:

```
POST /comment.php HTTP/1.1
Host: somesite.com
Content-length: 2352
Content-type: text/html
Accept: */*

user=John&comment=<script>alert('XSS')</script>
```

“Never play with JavaScript.”

5. Security Misconfigurations

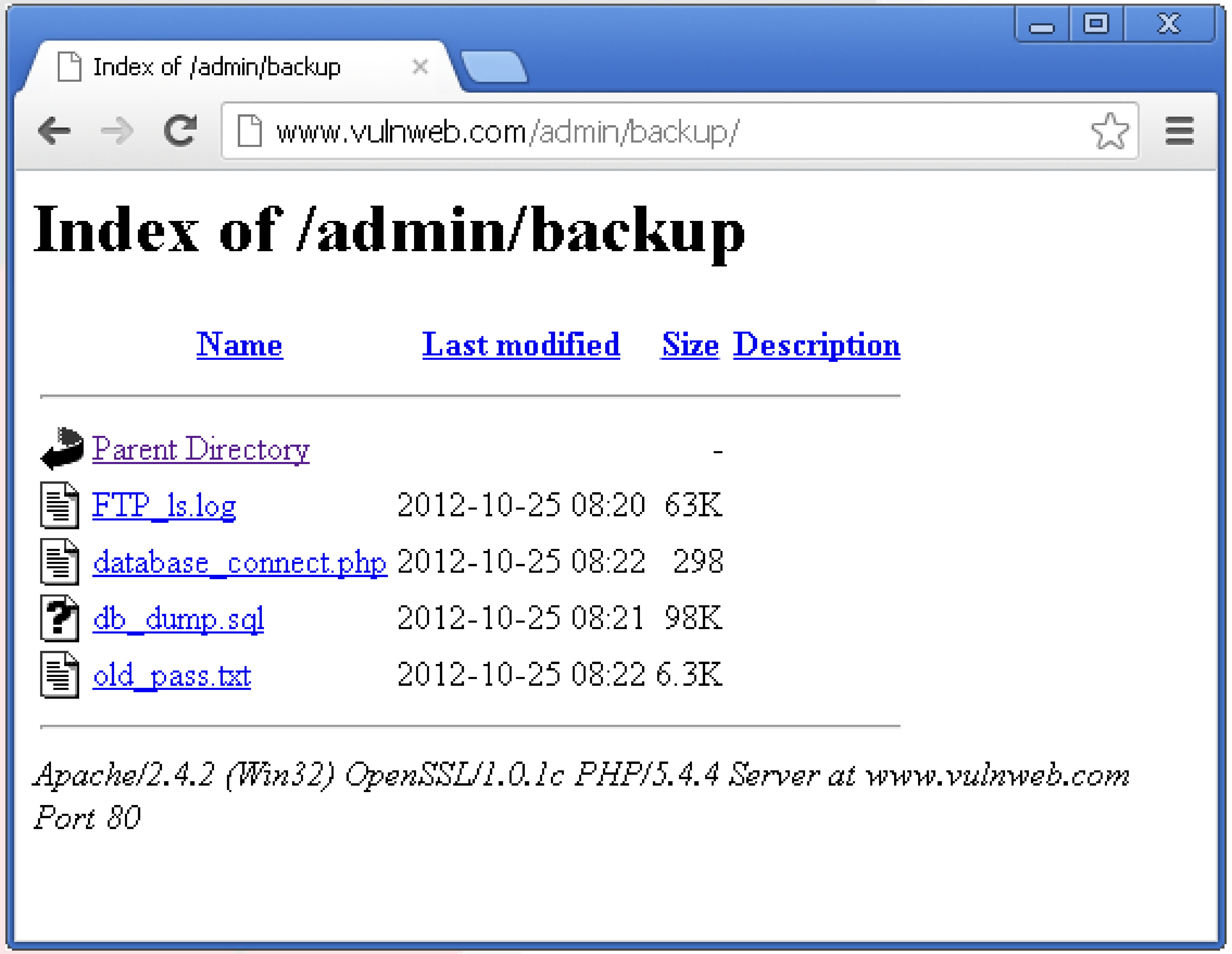
Security Misconfigurations

1. Not changing default passwords
   1. admin/admin
   2. admin/password
   3. root/password
   4. login/password

2. Directory access enabled

3. Debug mode enabled

4. Errors/stack traces



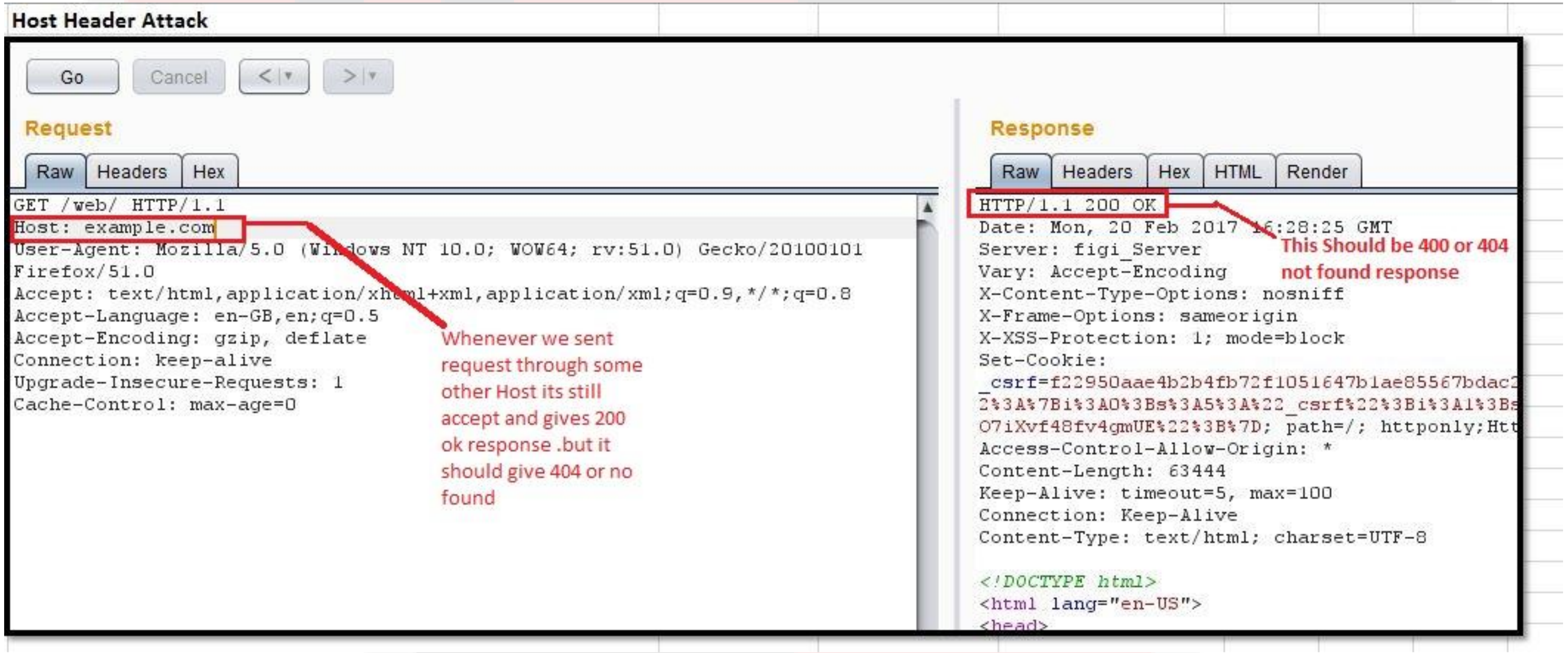
Index of /admin/backup

Name	Last modified	Size	Description
Parent Directory		-	
FTP_ls.log	2012-10-25 08:20	63K	
database_connect.php	2012-10-25 08:22	298	
db_dump.sql	2012-10-25 08:21	98K	
old_pass.txt	2012-10-25 08:22	6.3K	

Apache/2.4.2 (Win32) OpenSSL/1.0.1c PHP/5.4.4 Server at www.vulnweb.com Port 80

Security Misconfigurations

5. Unvalidated Host Header Injections



```
GET /web/ HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:51.0) Gecko/20100101
Firefox/51.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```

Whenever we sent request through some other Host its still accept and gives 200 ok response .but it should give 404 or no found

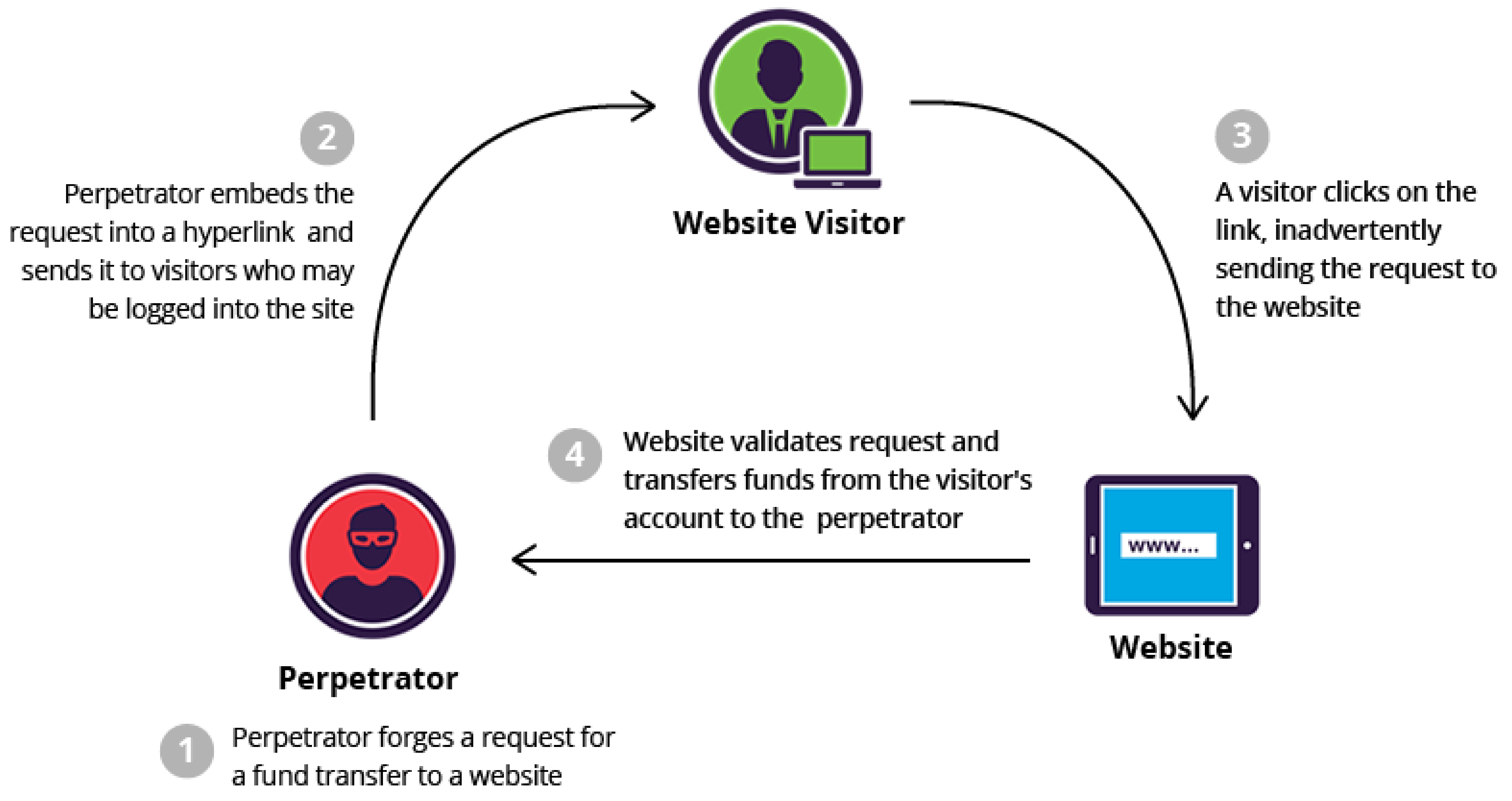
```
HTTP/1.1 200 OK
Date: Mon, 20 Feb 2017 16:28:25 GMT
Server: figi_Server
Vary: Accept-Encoding
X-Content-Type-Options: nosniff
X-Frame-Options: sameorigin
X-XSS-Protection: 1; mode=block
Set-Cookie:
_csrf=f22950aae4b2b4fb72f1051647b1ae85567bdac2
2%3A%7Bi%3A0%3Bs%3A5%3A%22_csrf%22%3Bi%3A1%3Bs
07iXvf48fv4gmUE%22%3B%7D; path=/; httponly;Htt
Access-Control-Allow-Origin: *
Content-Length: 63444
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

<!DOCTYPE html>
<html lang="en-US">
<head>
```

This Should be 400 or 404 not found response

“Did you check the closet.”

6. Cross Site Request Forgery (CSRF)

Cross Site Request Forgery


Website Visitor
Website
www...
Perpetrator
2
Perpetrator embeds the request into a hyperlink and sends it to visitors who may be logged into the site
3
A visitor clicks on the link, inadvertently sending the request to the website
4
Website validates request and transfers funds from the visitor's account to the perpetrator
1
Perpetrator forges a request for a fund transfer to a website

Cross Site Request Forgery

1. GET Scenario

```
GET http://bank.com/transfer.do?acct=BOB&amount=100 HTTP/1.1

<img src="http://bank.com/transfer.do?acct=MARIA&amount=100000"
width="0" height="0" border="0">
```

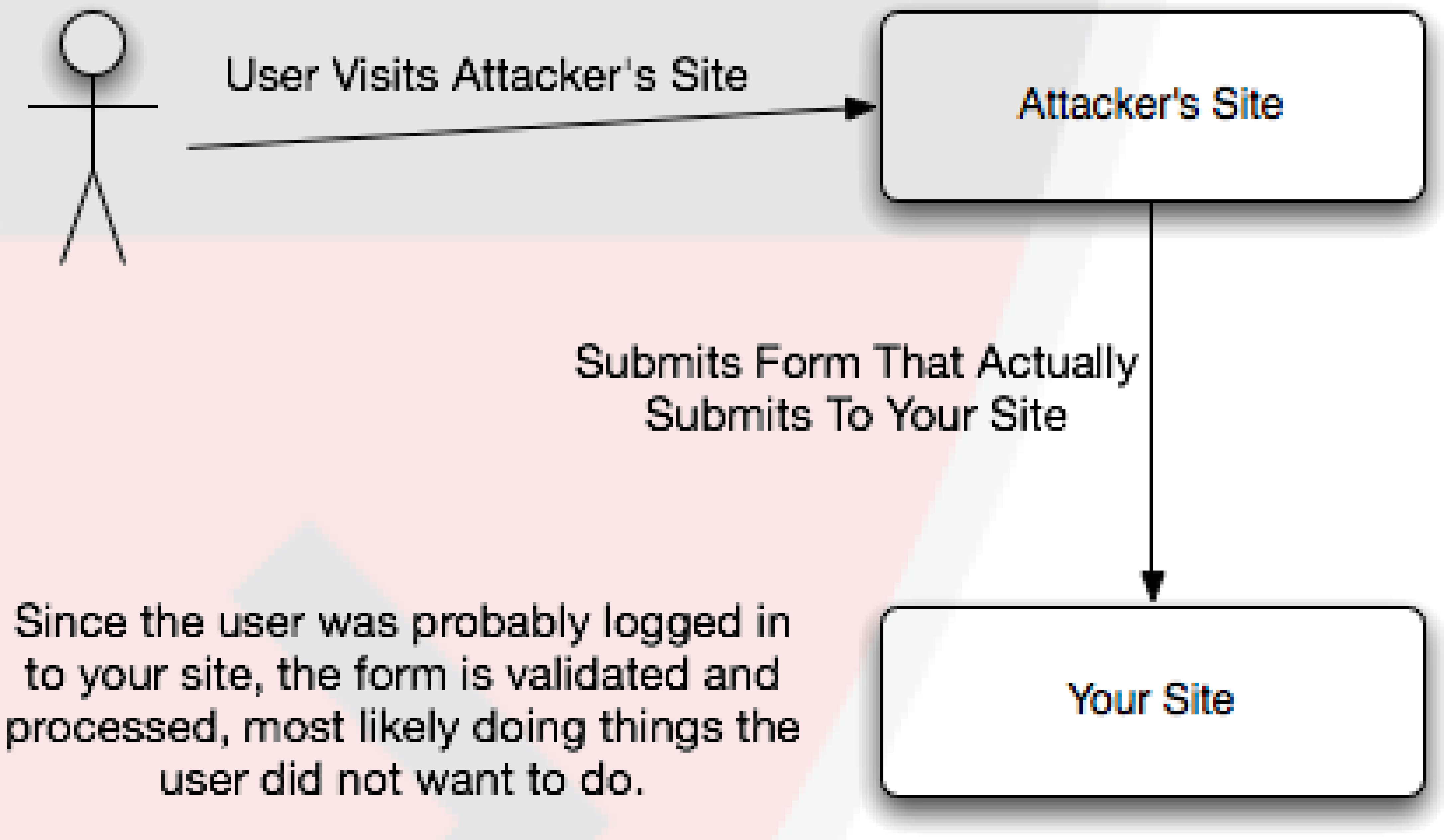
2. POST Scenario

```
POST http://bank.com/transfer.do HTTP/1.1

acct=BOB&amount=100
```

Cross Site Request Forgery

1. Causes:
   1. **No request tokens**
   2. **No per session tokens**
   3. **CORS Enabled**

   Access-Control-Allow-Origin: *


User Visits Attacker's Site
Attacker's Site
Submits Form That Actually Submits To Your Site
Since the user was probably logged in to your site, the form is validated and processed, most likely doing things the user did not want to do.
Your Site

“Attend the party without invitation.”

7. Unvalidated Redirects and Forwards

Unvalidated Redirects and Forwards

1. Some pages use a parameter to indicate where the user should be sent if a transaction is successful.

   http://www.example.com/boring.jsp?fwd=student.jsp

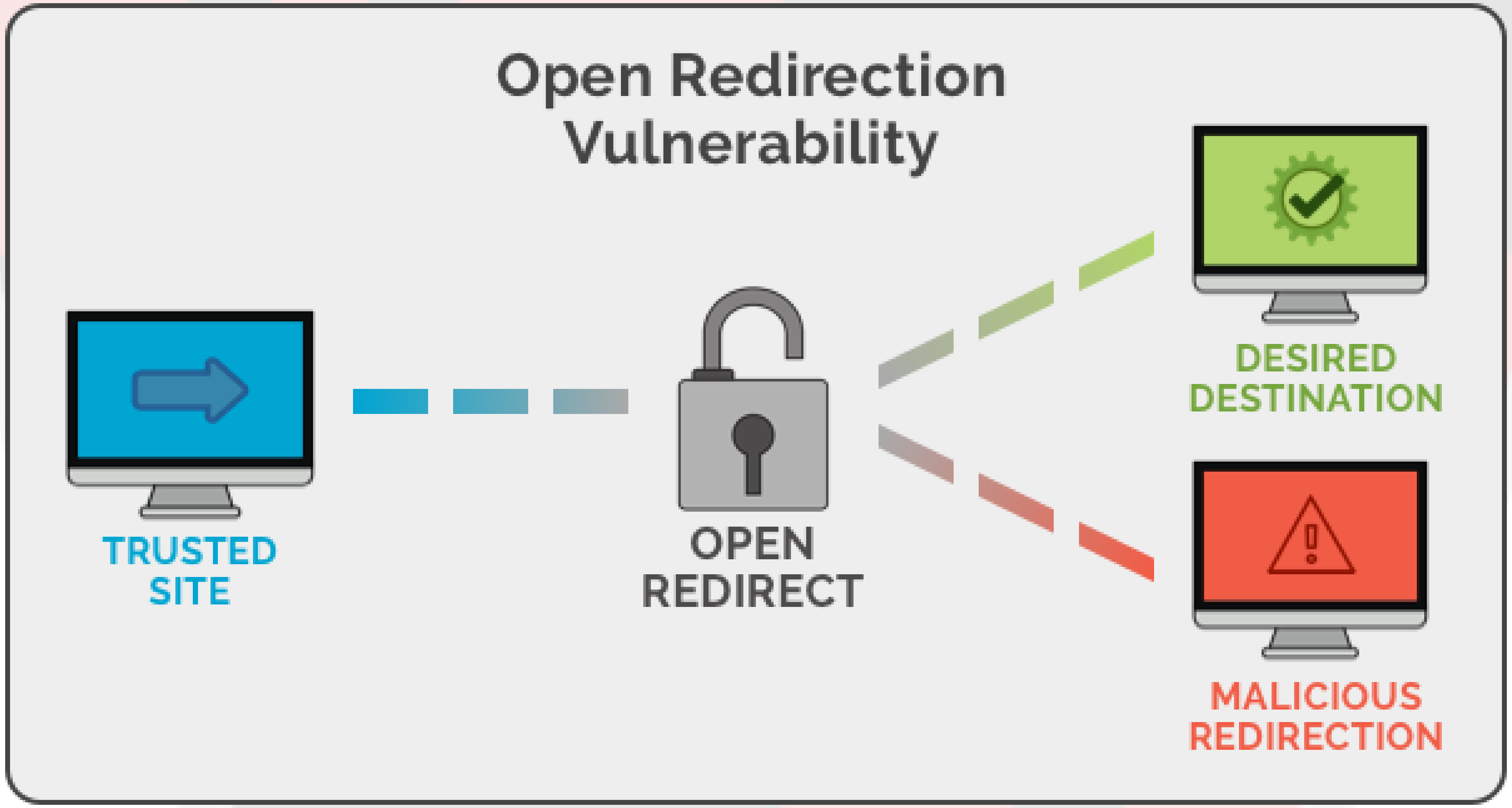
   http://www.example.com/boring.jsp?fwd=admin.jsp

2. Open Redirects

   https://example.com/redirect.php

   https://example.com/redirect.php?go=http://attacker.com/phish/

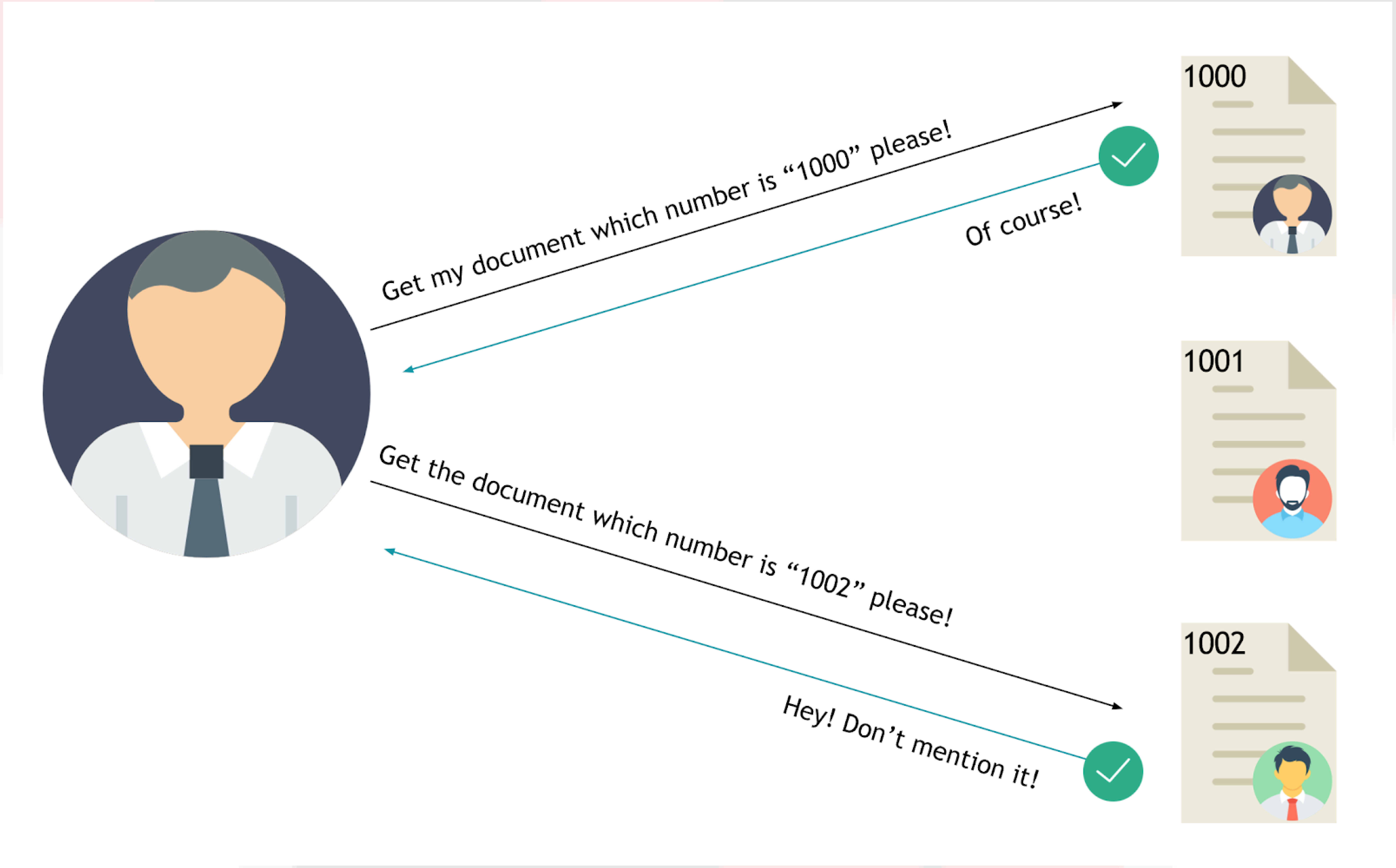
Unvalidated Redirects and Forwards



“Don’t flow with the Internet.”

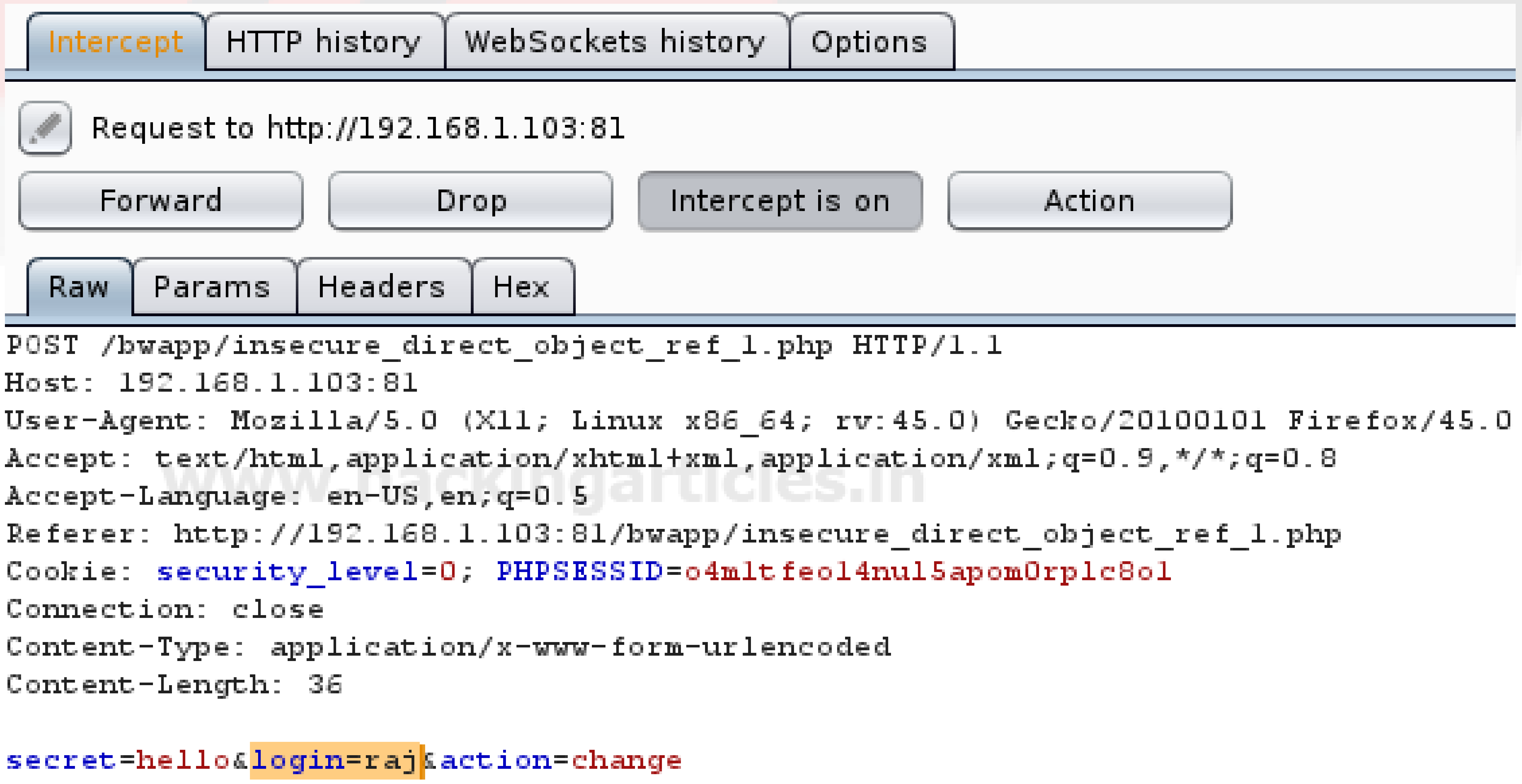
8. Insecure Direct Object Reference (IDOR)

Insecure Direct Object Reference


Get my document which number is "1000" please!
Of course!
Get the document which number is "1002" please!
Hey! Don't mention it!
1000
1001
1002

Insecure Direct Object Reference

1. When a reference to an internal implementation object, such as a file or database key, is exposed to users



```
POST /bwapp/insecure_direct_object_ref_1.php HTTP/1.1
Host: 192.168.1.103:81
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gecko/20100101 Firefox/45.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: http://192.168.1.103:81/bwapp/insecure_direct_object_ref_1.php
Cookie: security_level=0; PHPSESSID=o4mltfeol4nul5apom0rplc8ol
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 36

secret=hello&login=raj&action=change
```

“Don’t let users spoil your abstraction.”

Nothing is 100% Foolproof

Understand 01

Assess 02

Plan 03

Implement 04

GET Queries?

Thank You!