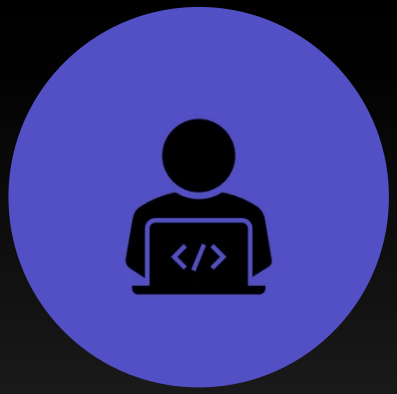
CROSS-SITE SCRIPTING (XSS)

UOM INFORMATION SECURITY CLUB

accenture

What is XSS?

CODE INJECTION ATTACK



JAVASCRIPT IS INJECTED INTO AN OTHER USER'S BROWSER

JavaScript? Malicious?

JS has access to sensitive information such as cookies

JS can make arbitrary HTTP requests

JS can be used to make arbitrary modifications to the HTML of the current page

How JavaScript Is Injected?

```
print "<html>"
print "Latest comment:"
print database.latestComment
print "</html>"
```

```
<html>
Latest comment:
<script>...</script>
</html>
```

Consequences of XSS:

Cookie theft

- access the victim's cookies
- extract sensitive information like session IDs.

Keylogging

- register a keyboard event listener
- Recording sensitive information such as passwords and credit card numbers.

Phishing

- insert a fake login form
- trick the user into submitting sensitive information.

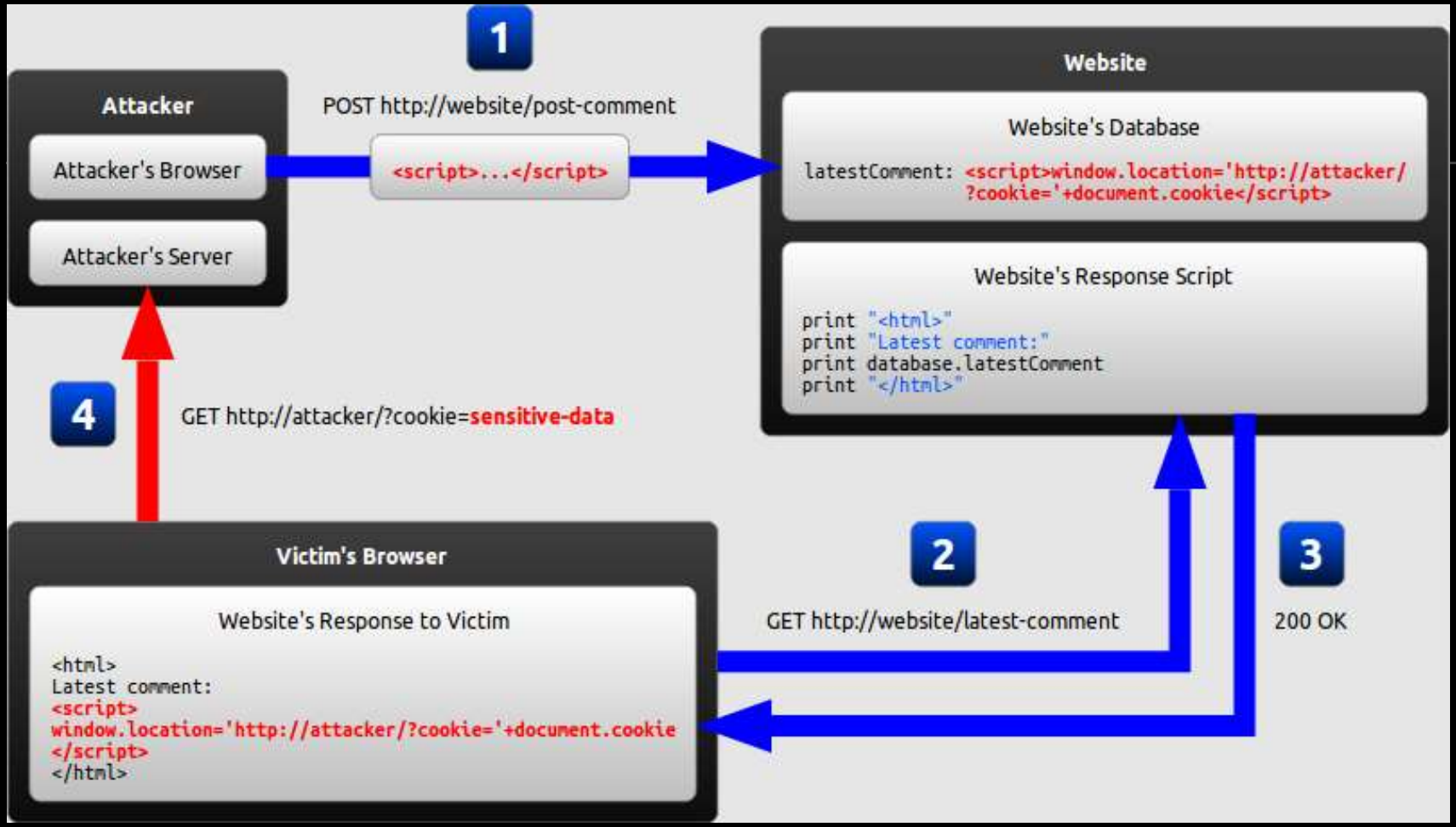
Types of XSS

Persistent XSS : malicious string originates from the website's database.

Reflected XSS : malicious string originates from the victim's request.

DOM-based XSS : vulnerability exists in the client-side code rather than the server-side.

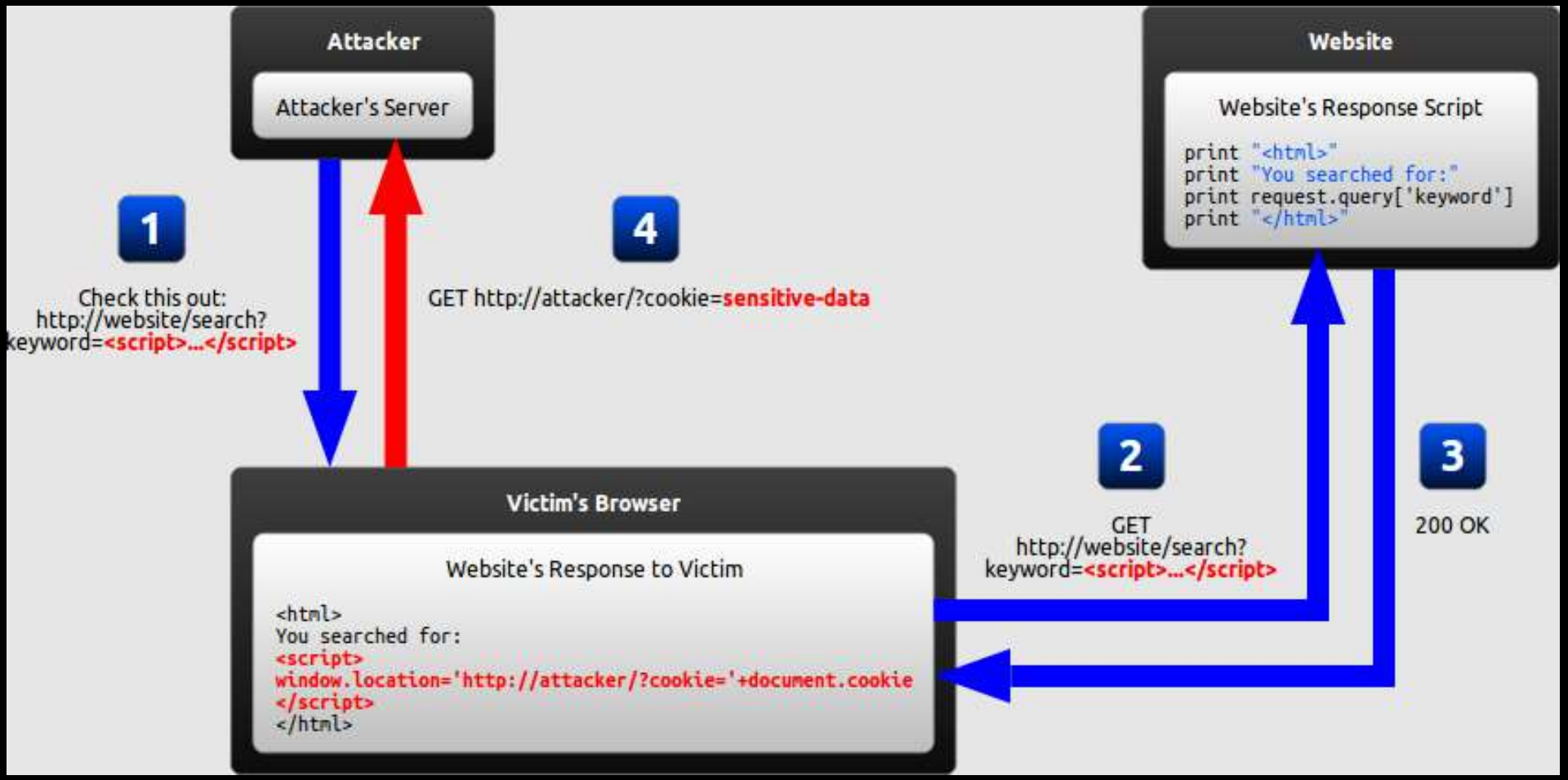
Example of Persistent XSS



Step-by-Step Analysis

1) The attacker uses one of the website's forms to insert a malicious string into the website's database.
2) The victim requests a page from the website.
3) The website includes the malicious string from the database in the response sent to the victim
4) The victim's browser executes the malicious script inside the response, sending the victim's cookies to the attacker's server.

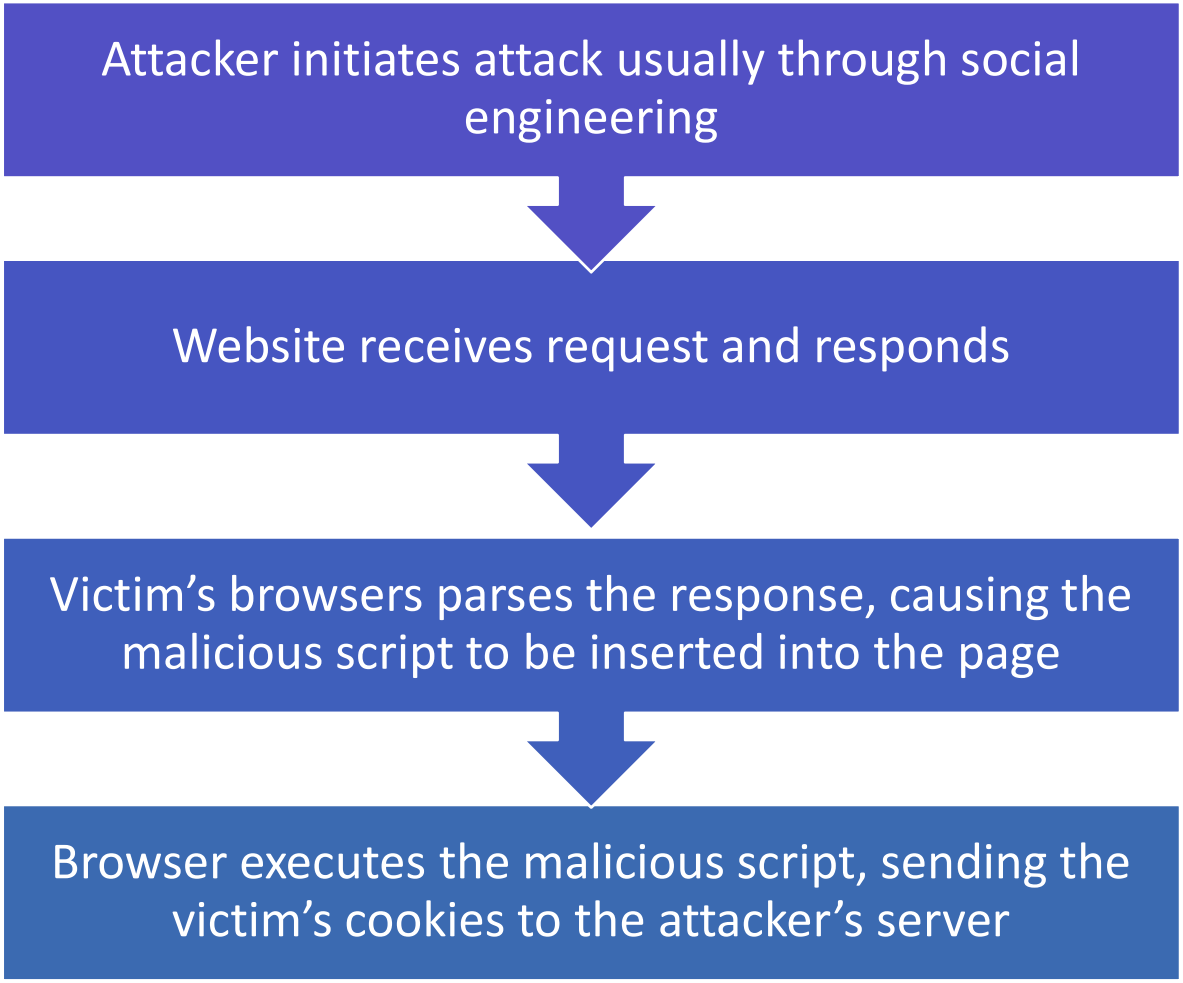
Closer Look at Reflected XSS Attacks



Step-by-Step Analysis

1. The attacker crafts a URL containing a malicious string and sends it to the victim.
2. The victim is tricked by the attacker into requesting the URL from the website.
3. The website includes the malicious string from the URL in the response.
4. The victim's browser executes the malicious script inside the response, sending the victim's cookies to the attacker's server.

DOM Based XSS
(Invisible to the server)



Preventing XSS

Encoding : escape user input so that the browser interprets it only as data, not as code.

Validation : filters user input so that the browser interprets the code without the malicious commands.

Time To Hack!

XSS Challenge:
https://xss-game.appspot.com/

Helpful starting XSS & JavaScript starter guide:
http://securityidiots.com/Web-Pentest/XSS/xss.series-by-securityidiots.html

Workshop slides and more resources:
https://github.com/umisc/workshops/tree/master/XSS