Security? Why bother?

Security? Why bother?

- Who am I?
- Why am I here?

Security? Why bother?

- Consequences of an insecure website
  - loss of business
  - Destroy customer confidence and brand
  - Legal liability
  - Financial loss
  - Costs of incident handling

Security? Why bother?

- WhiteHat security research - all data collected through vulnerability assessment of the largest and most popular websites in the retail, financial, insurance, education and social networks.
- Study uses WASC threat classification

Security? Why bother? Because people are evil!

- Authentication
  - Brute Force
  - Insufficient Authentication
- Authorization
  - Insufficient Authorization

Security? Why bother? Because people are evil!

- Client-side attacks
  - Cross site scripting - XSS
  - Cross site request forgery - CSRF

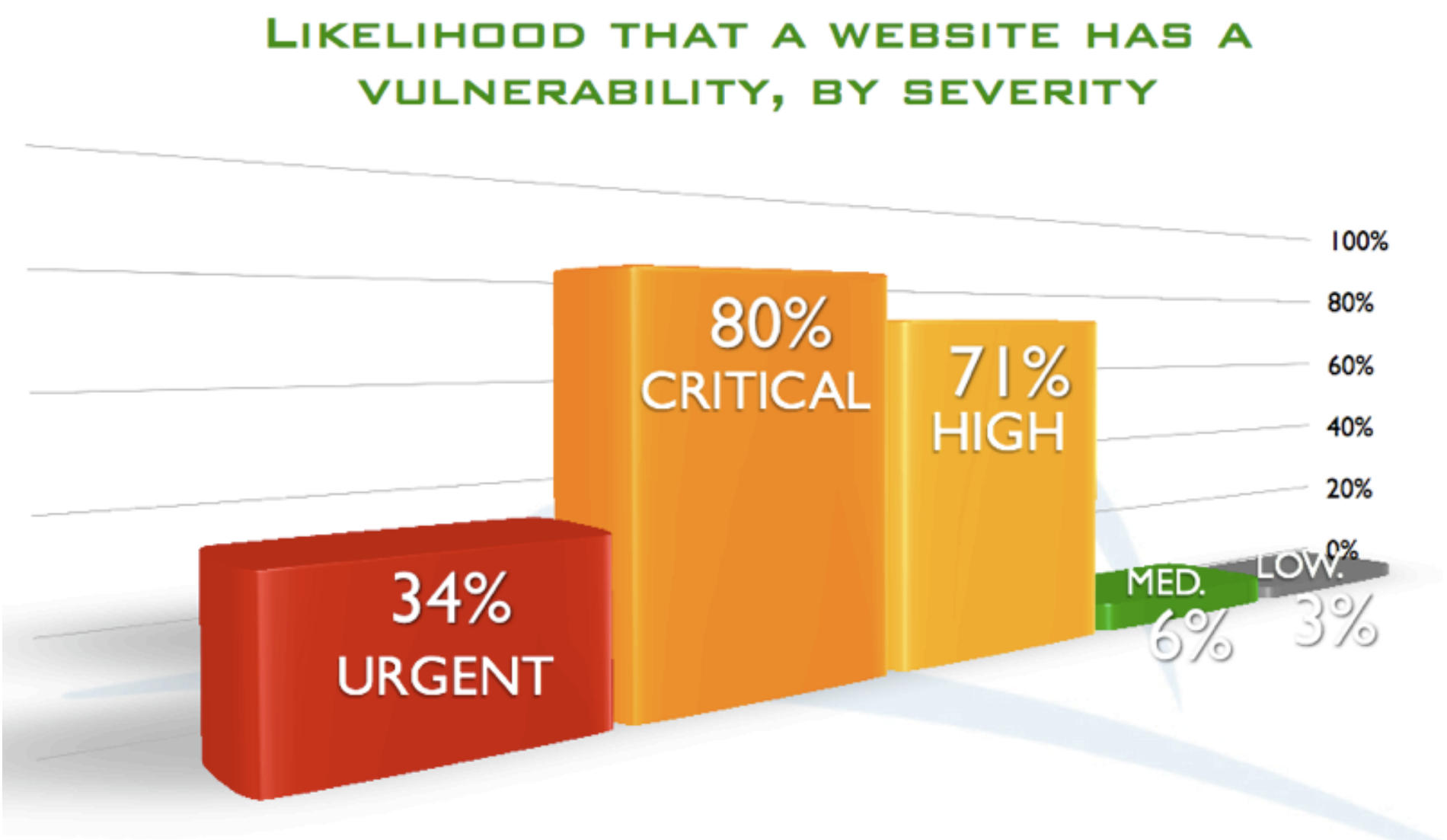
Security? Why bother? Because people are evil!

- Command Execution
  - SQL Injection
- Information Disclosure
  - Information Leakage
  - Directory Indexing
  - Predictable Resource Location

Security? Why bother?

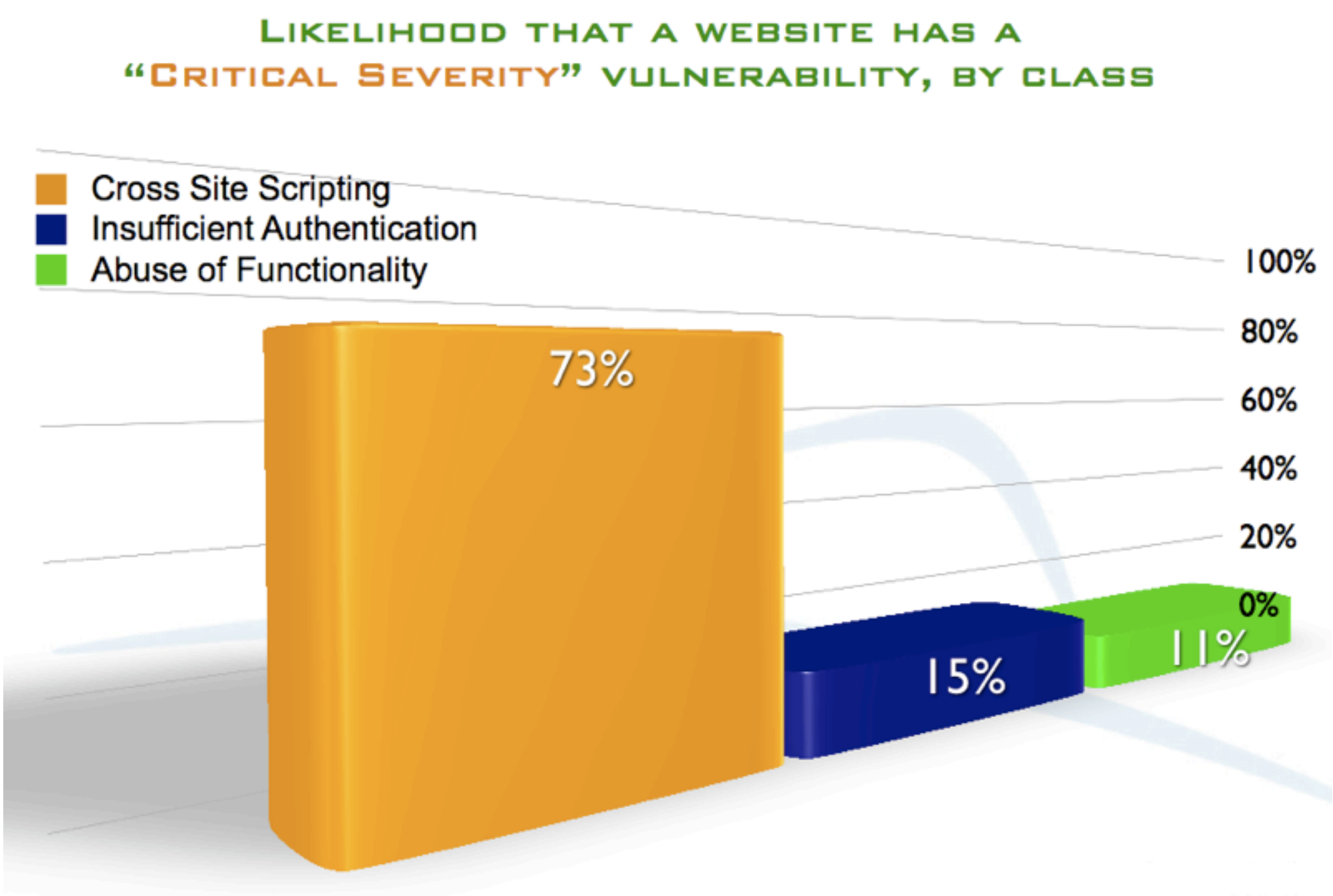
7 out of 10 websites have serious vulnerabilities

But how bad is it really?

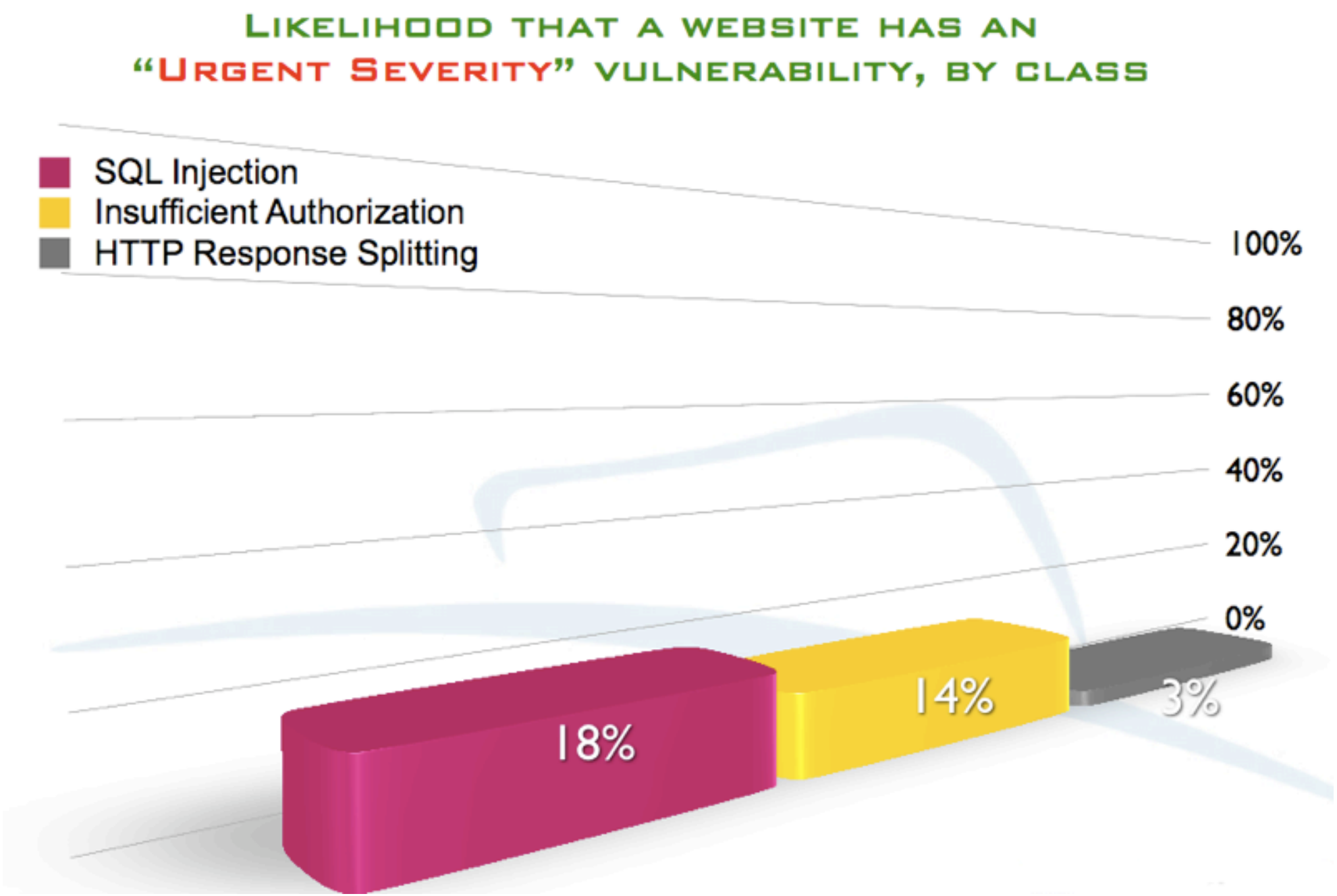


Websites with Urgent, Critical, or High severity issues technically would not pass PCI compliance

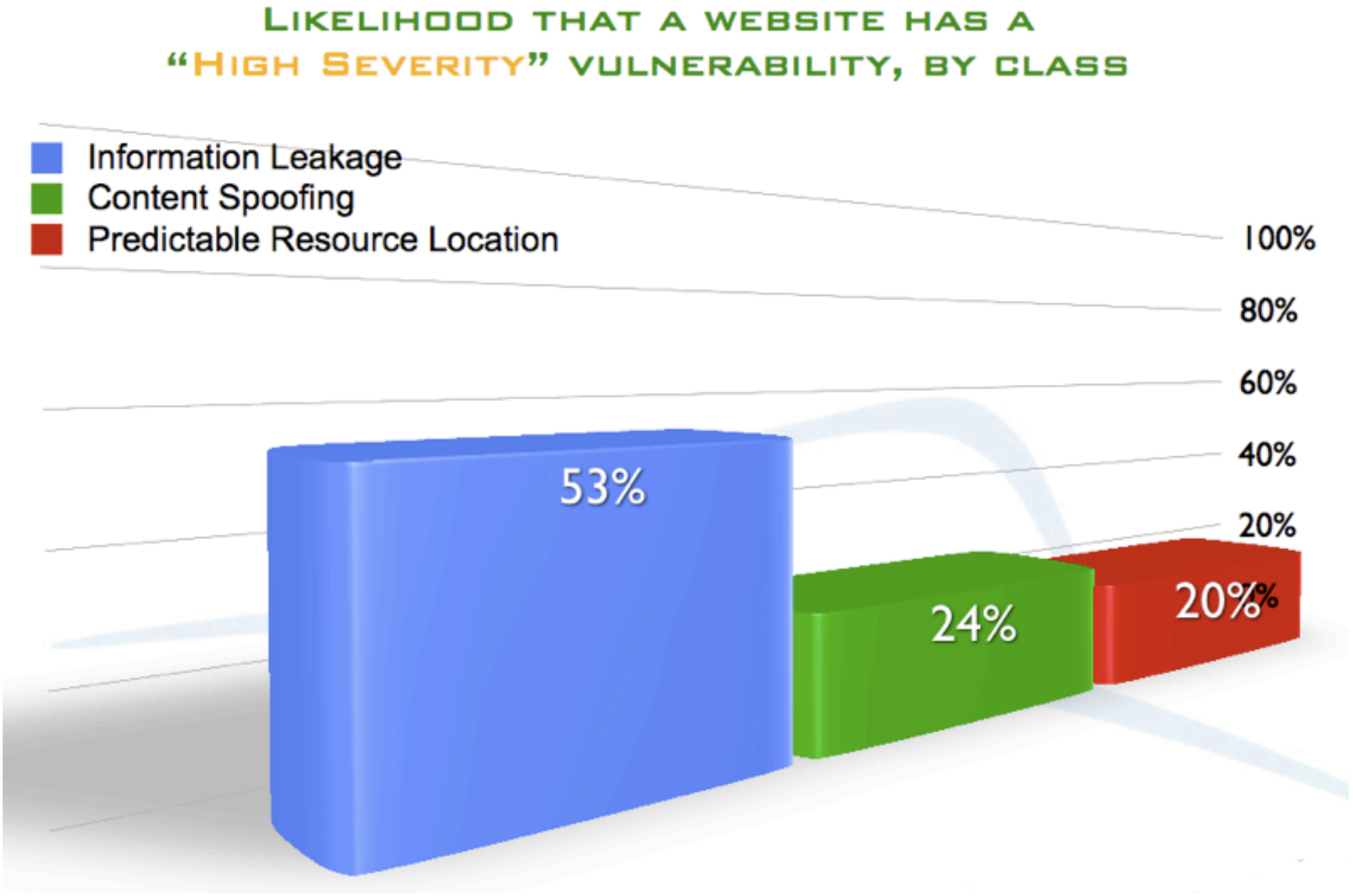
Critical


Likelihood that a website has a "Critical Severity" vulnerability, by class
Cross Site Scripting
Insufficient Authentication
Abuse of Functionality
100%
80%
60%
40%
20%
0%
73%
15%
11%

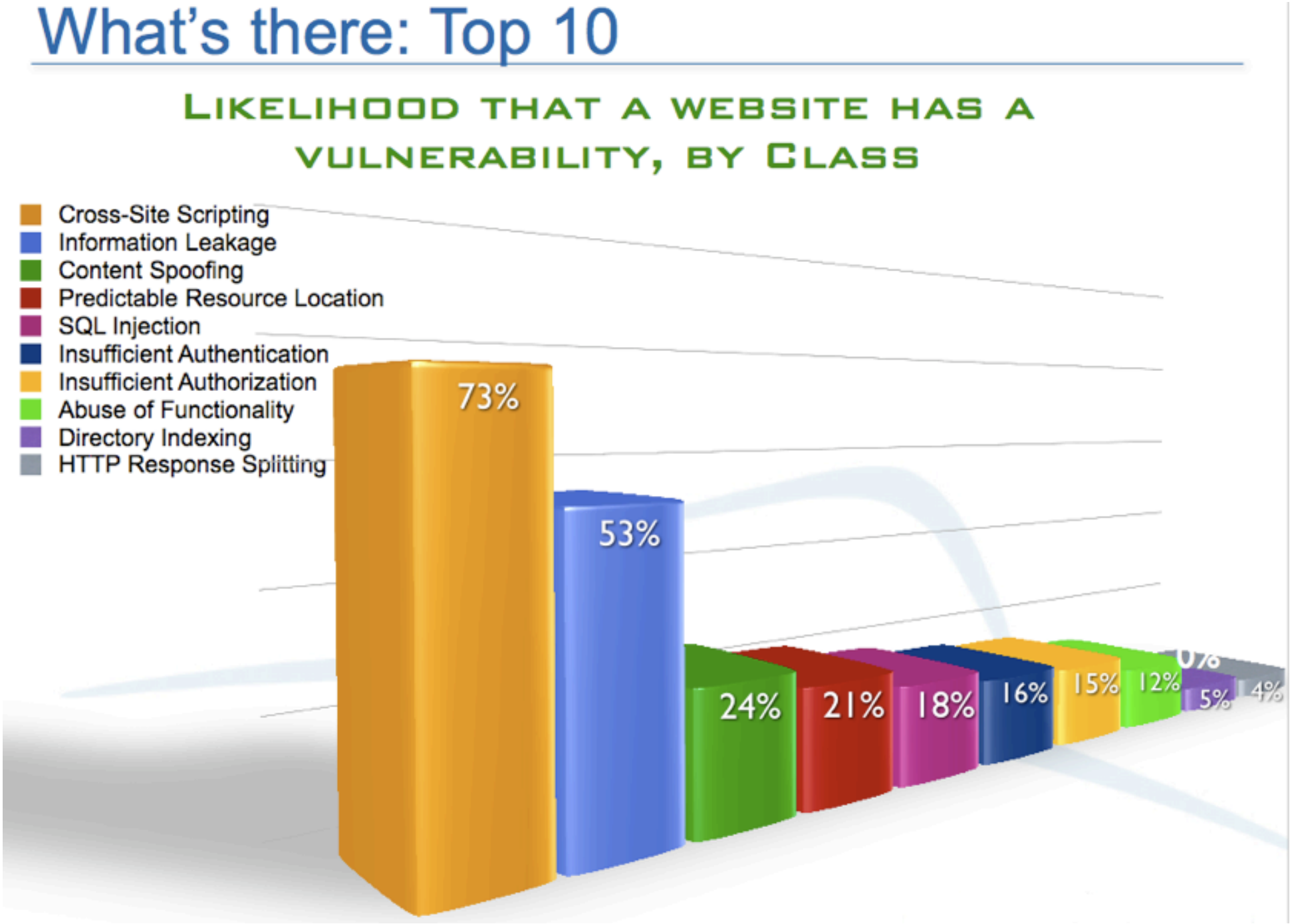
Urgent


Likelihood that a website has an "Urgent Severity" vulnerability, by class
SQL Injection
Insufficient Authorization
HTTP Response Splitting
100%
80%
60%
40%
20%
0%
18%
14%
3%

High

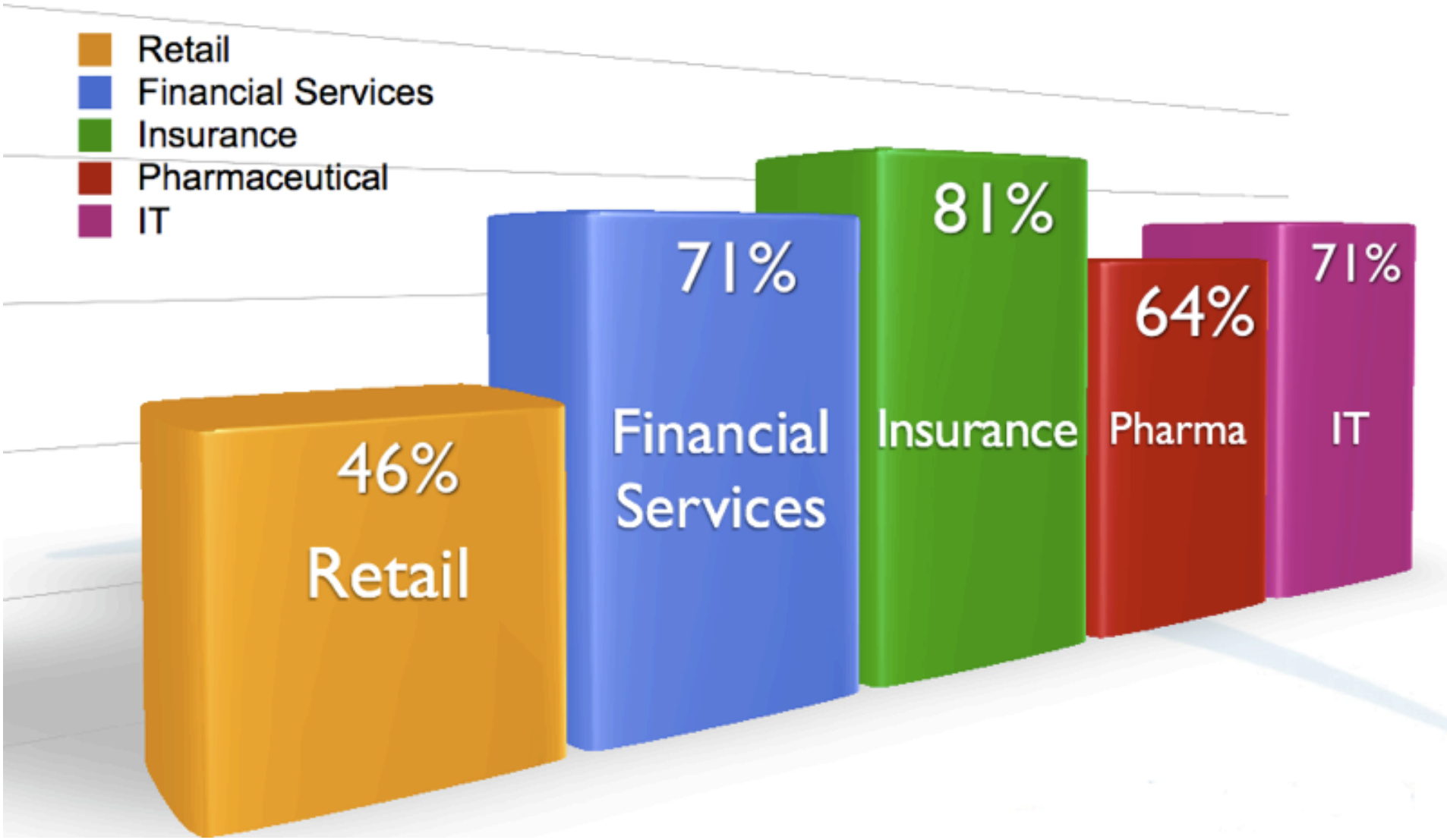

Likelihood that a website has a
"High Severity" vulnerability, by class
Information Leakage
Content Spoofing
Predictable Resource Location
100%
80%
60%
40%
20%
53%
24%
20%

What's there: Top 10



Comparing industry verticals

Likelihood that a website in a particular has a vulnerability (at least 25 websites)



Security? Why bother? Because people are evil!

- What can you do?
- Turn off your server? No, but you can place as much obstacles as possible.
- Asset tracking - you cannot secure what you do not know you own. Install the update status module.
- Vulnerability assessment (measure security) - Use tools such as tamper data, webscarab or any interception proxy, nikto, port scanners.

Security? Why bother? Because people are evil!

If you are a developer

- Read the "writing secure code" handbook (http://drupal.org/writing-secure-code). ask questions in the forum or #drupal IRC channel
- Local OWASP chapters, OWASP mailing list, bugtrack, gnucitzen.org, wasc webappsec.org, planet-websecurity.org

Drupal Anti-SQL injection

- NO: `db_query("SELECT * FROM {table} WHERE someval = '$user_input'");`
- YES: `db_query("SELECT * FROM {table} WHERE someval = '%s'", $user_input);`

Drupal Anti-XSS

t()

```
t('I escape %user_data', array('%user_data' => $data));
  I escape <em>user_data</em> (safe)

t('I escape @user_data', array('@user_data' => $data));
  I escape user_data (safe)

t('I do not escape !user_data', array('!user_data' => $data));
  XSS vulnerability
```

check_plain - to be used when inserting plain text in HTML

check_markup - to be used when inserting rich text in HTML

filter_xss - to remove all but whitelisted tags from text inserted in HTML
filter_xss_admin - shortcut to filter_xss with a permissive tag list, used to output admin defined texts.

Drupal's FAPI

- Valid choice checker
- Protects against Cross site request forgeries