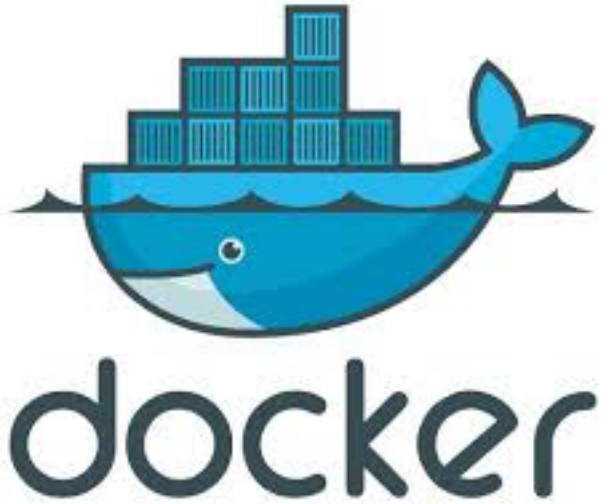
Cross Site Scripting

To alert() and beyond

What?

- Introduction to XSS
- How to find XSS
- What can be done
- Tools to help exploit XSS
- Defense against XSS

- Lots of Demos
  - Play Along – https://github.com/jbarone/xsslab



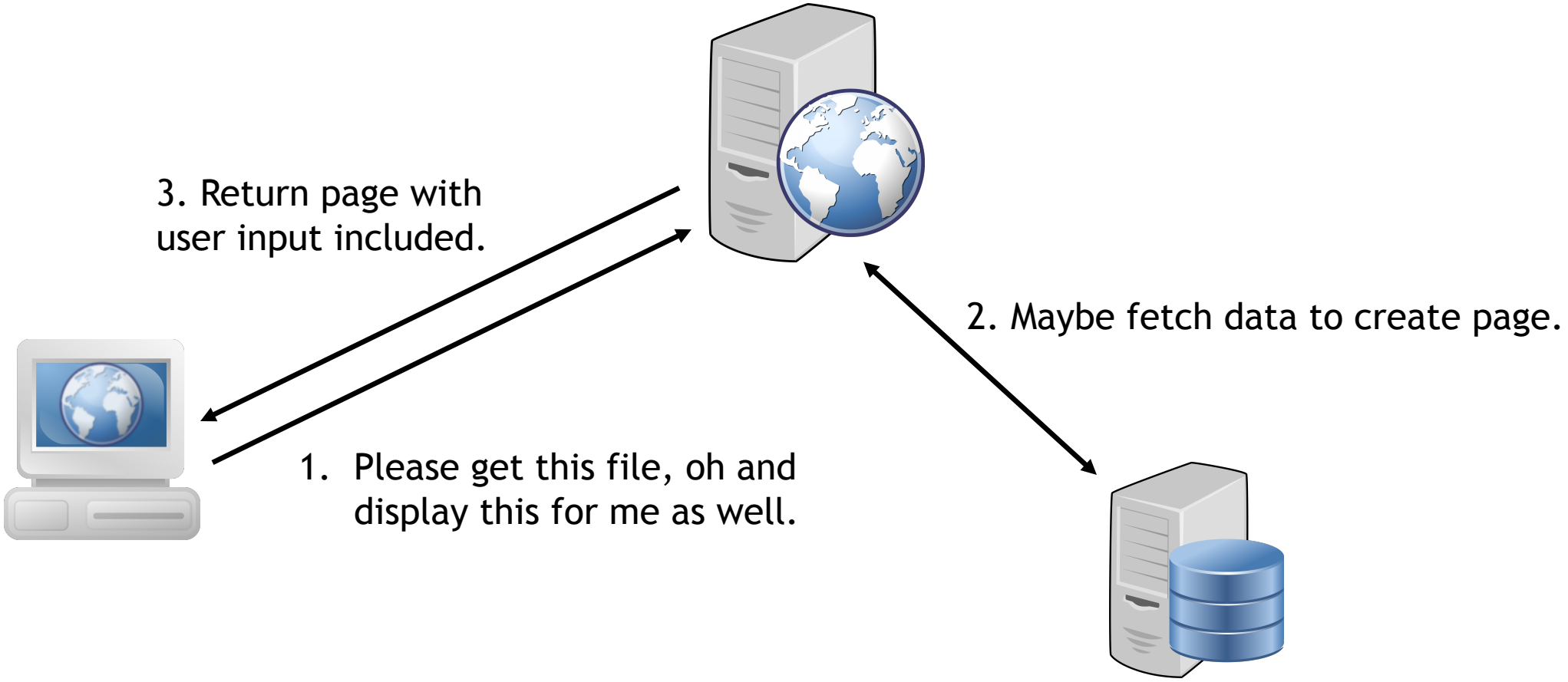
Nope

- How to code JavaScript
- Single Origin Policy Bypass
- Browser evasion
- Server evasion
- How to hack the Gibson

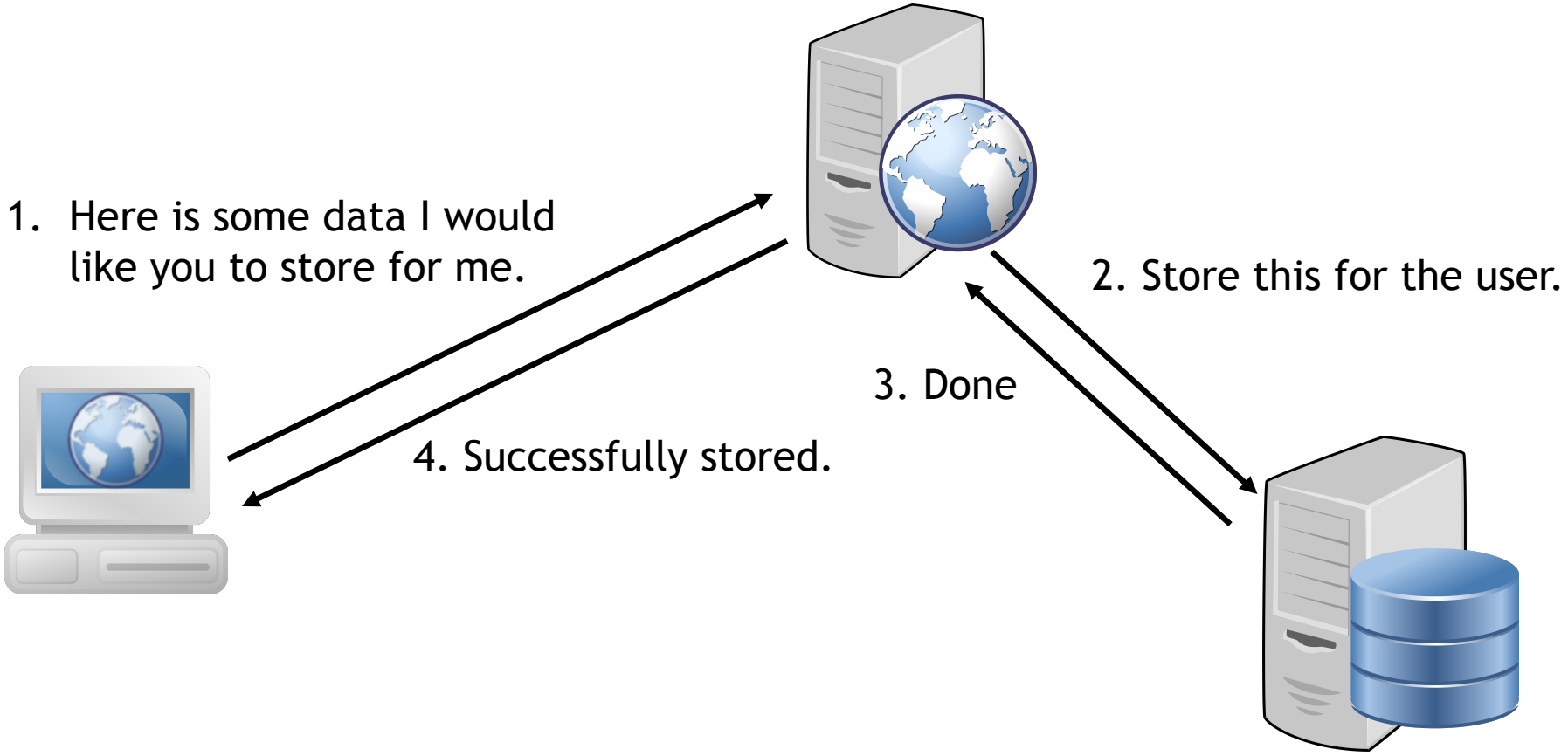
Cross site scripting

Classifications of Attacks

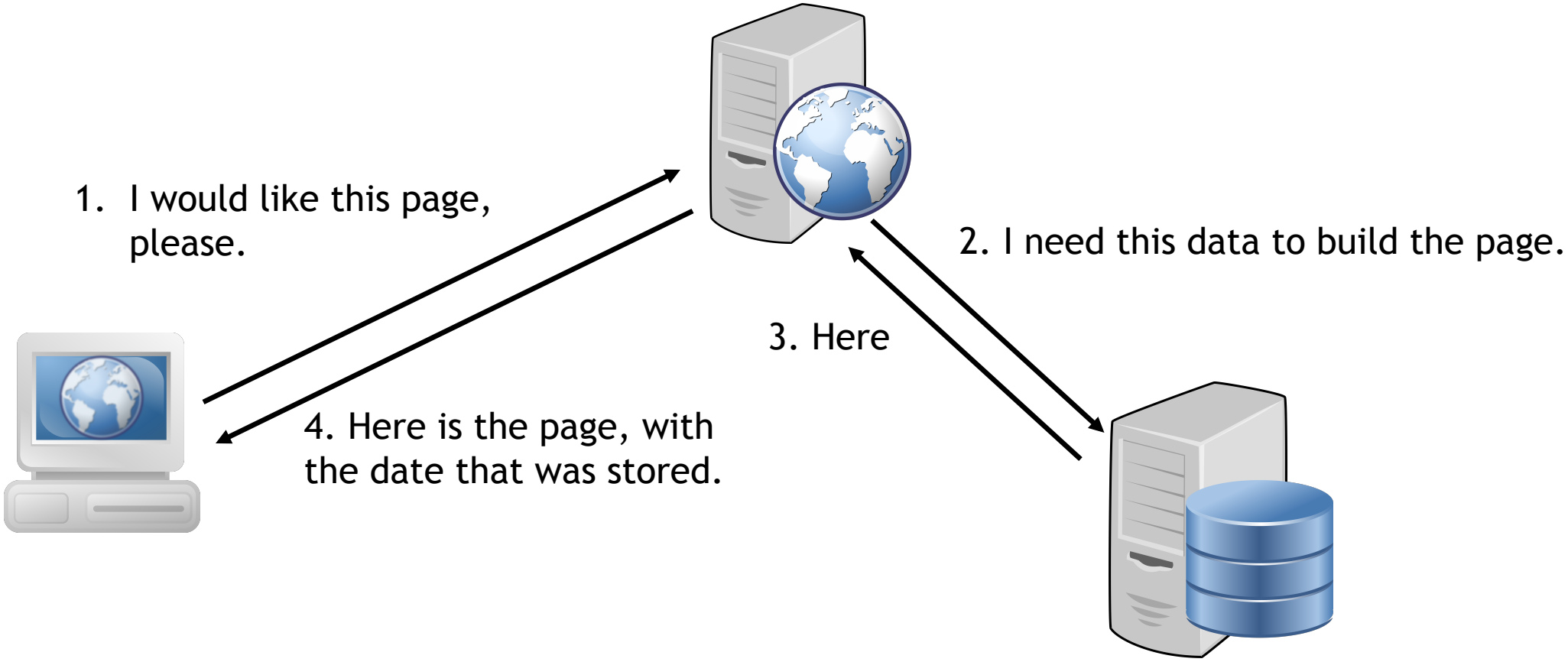
Reflected


3. Return page with user input included.
2. Maybe fetch data to create page.
1. Please get this file, oh and display this for me as well.

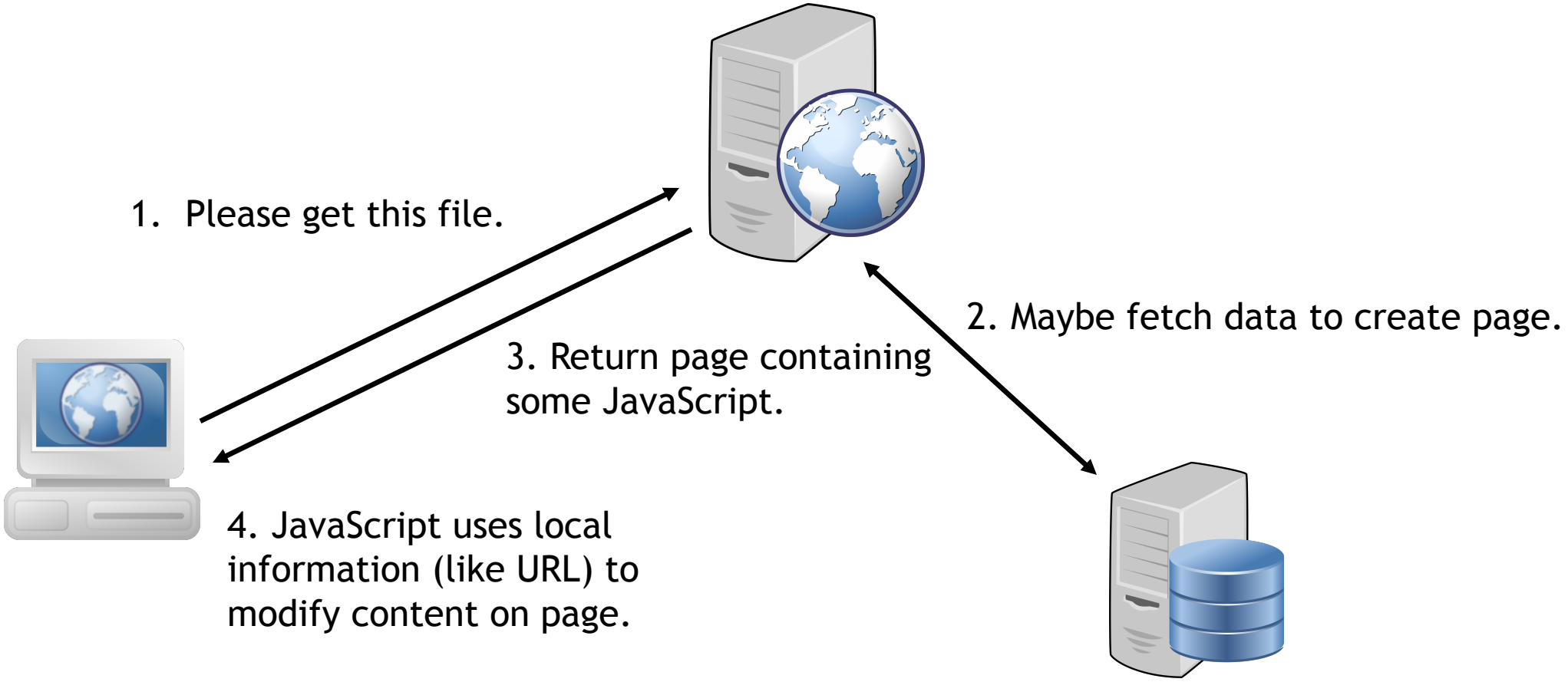
Persistent


1. Here is some data I would like you to store for me.
2. Store this for the user.
3. Done
4. Successfully stored.

Persistent (cont.)



DOM-Based


1. Please get this file.
2. Maybe fetch data to create page.
3. Return page containing some JavaScript.
4. JavaScript uses local information (like URL) to modify content on page.

DEMO TIME!!

Finding XSS

Context matters

- HTML Injection
- Inside HTML Tag
  - IMG tag src
  - Event attribute tags (onerror)
- Inside JavaScript code block

Tools

- XSSSNIPER – https://github.com/gbrindisi/xsssniper
- XSSER (has a GUI) - https://xsser.03c8.net/
- XSSCRAPY - https://github.com/DanMcInerney/xsscrapy
- Interception Proxies
  - ZAP - https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project
  - Burp - https://portswigger.net/burp/

The Attacks

More than alert()

Attacking Users

- Deface Content
- Capture Input
- Social Engineering
  - Get user to submit passwords
  - Get user to install “software”
- Session stealing
- Session Riding (CSRF)

Attacking Browsers

- Sandbox Escapes
- Heap Exploitation
- Attacks on Extensions
- Attacks on Plugins
- Proxy through the browser
- Metasploit attacks

Attacking Networks

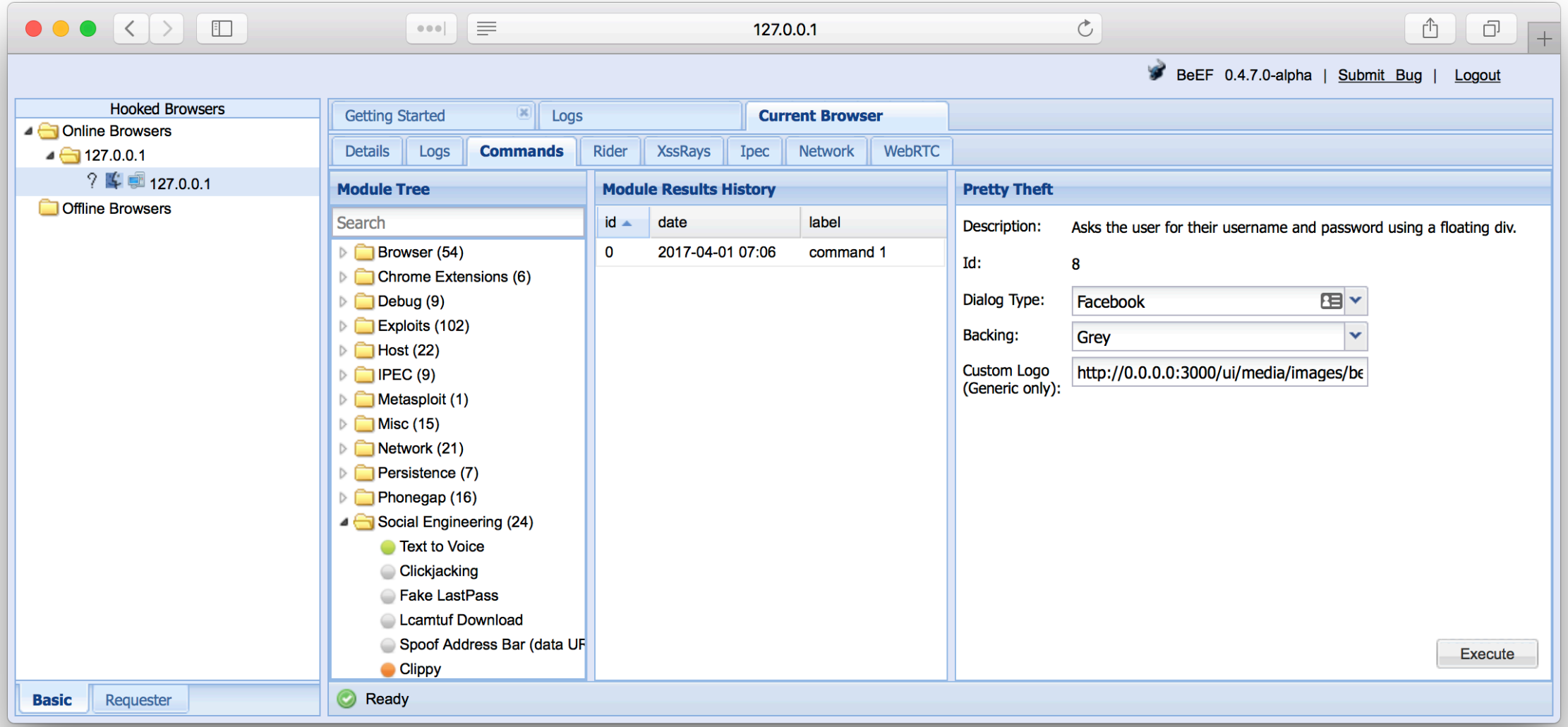
- Ping Sweeps
- Port Scanning
- Fingerprinting services
- Attacking services

DEMO TIME!!

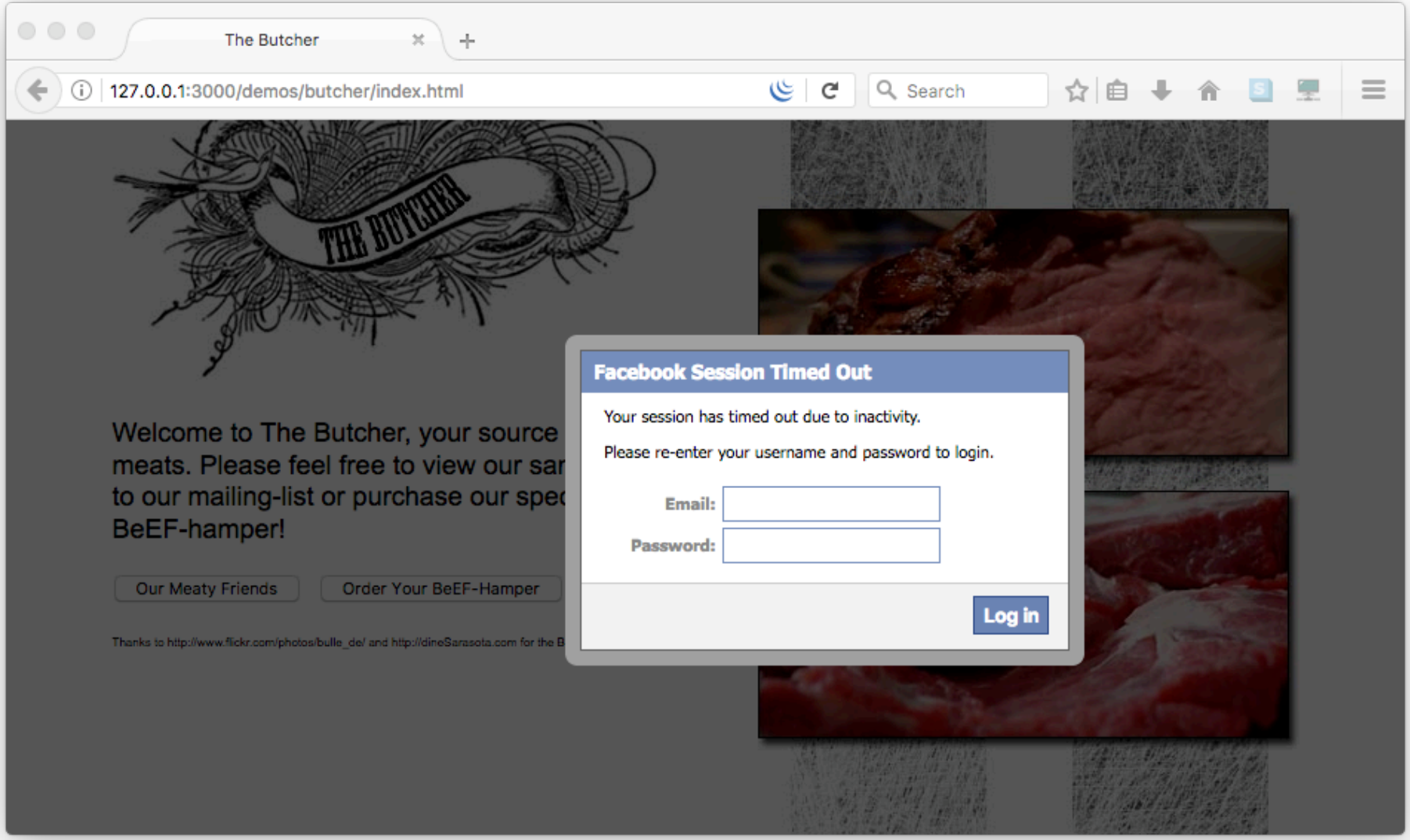
BeEF

- Browser Exploitation Framework
- Designed to exploit XSS
- Focus on payload delivery
- Written in Ruby
- Integrates with Metasploit

BeEF (Continued)



BeEF (Continued)



Defense

Defense

- Developers
  - **NEVER TRUST USER INPUT!!!!!!!**
    - Sanitize all input (server and client)
    - Sanitize all output
- Users
  - Use browsers with XSS protections
    - Chrome
    - Edge
  - Look before you Click
    - Expand shortened URLs

Thanks

- Joshua Barone
- Twitter: @tygarsai
- Email: joshua.barone@gmail.com
- Blog: http://caveconfessions.com
- Lab: https://github.com/jbarone/xsslab.git