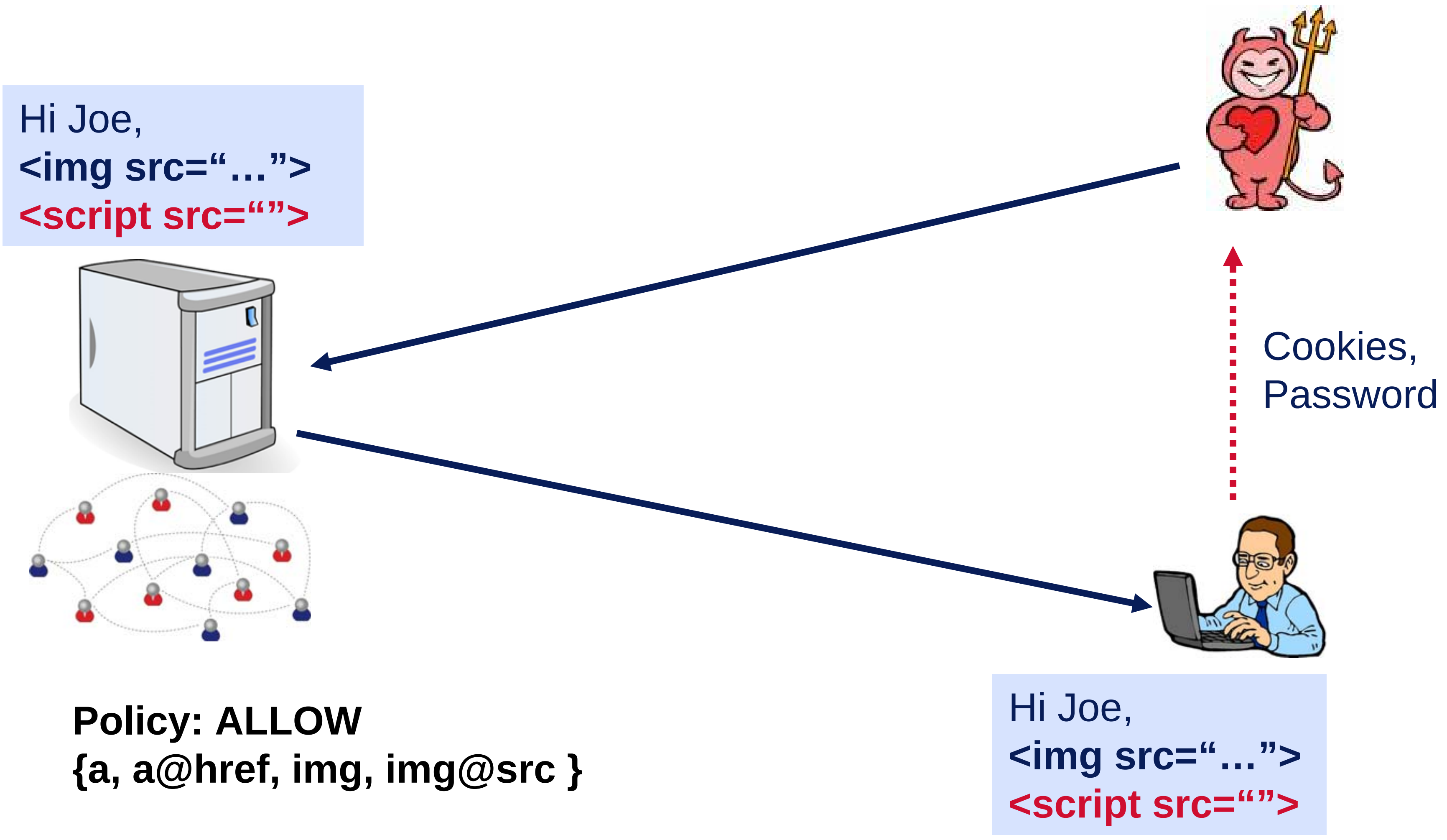
Document Structure Integrity: A Robust Basis for Cross-Site Scripting Defense

Yacin Nadji
Illinois Institute Of Technology

Prateek Saxena
UC Berkeley

Dawn Song
UC Berkeley

A Cross-Site Scripting Attack



Hi Joe,
<img src="…">
<script src="">

Policy: ALLOW
{a, a@href, img, img@src }

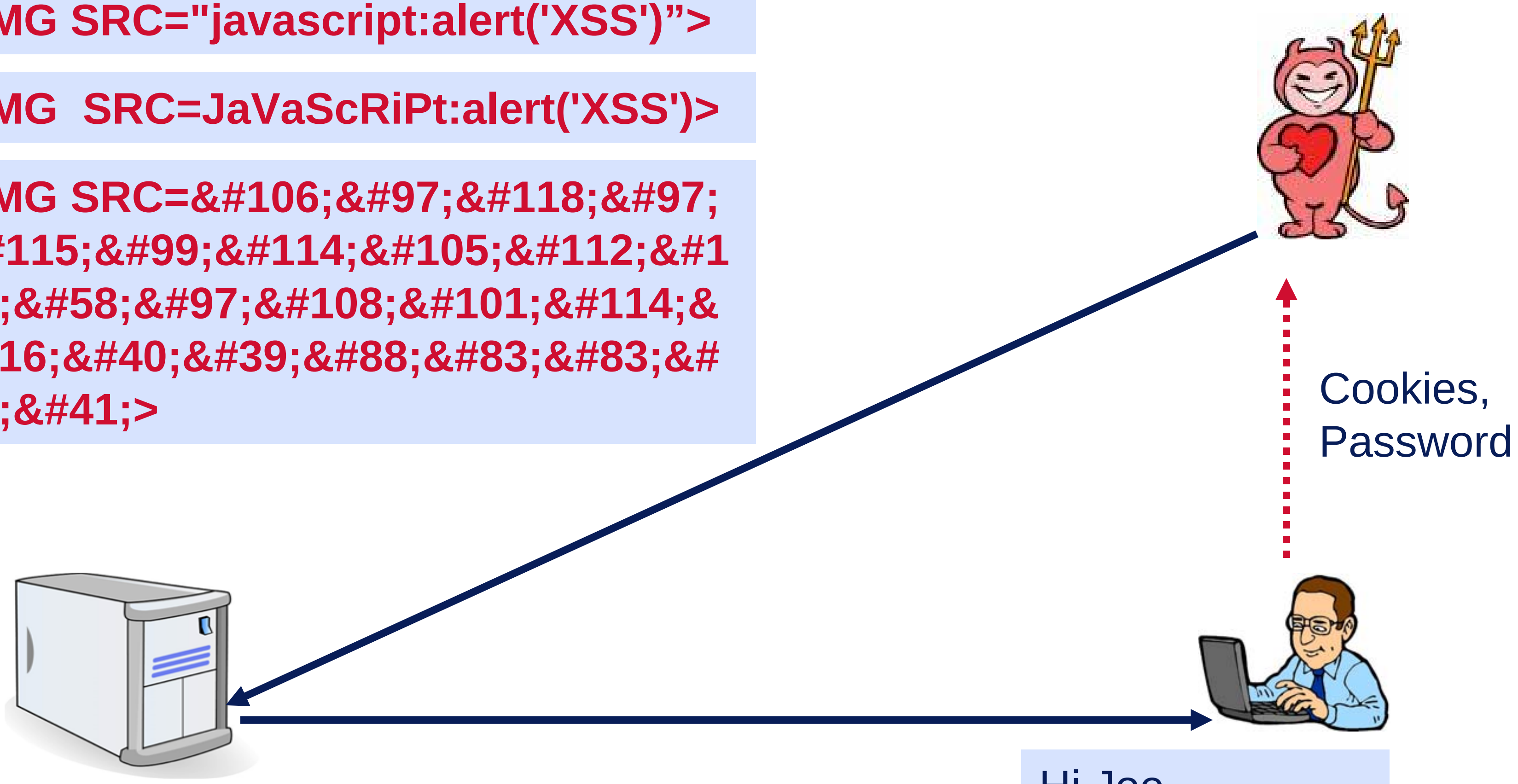
Hi Joe,
<img src="…">
<script src="">

Limitations of Server-side Sanitization

```
<IMG SRC="javascript:alert('XSS')”>
```

```
<IMG  SRC=JaVaScRiPt:alert('XSS')>
```

```
<IMG SRC=&#106;&#97;&#118;&#97;
&#115;&#99;&#114;&#105;&#112;&#1
16;&#58;&#97;&#108;&#101;&#114;&
#116;&#40;&#39;&#88;&#83;&#83;&#
39;&#41;>
```

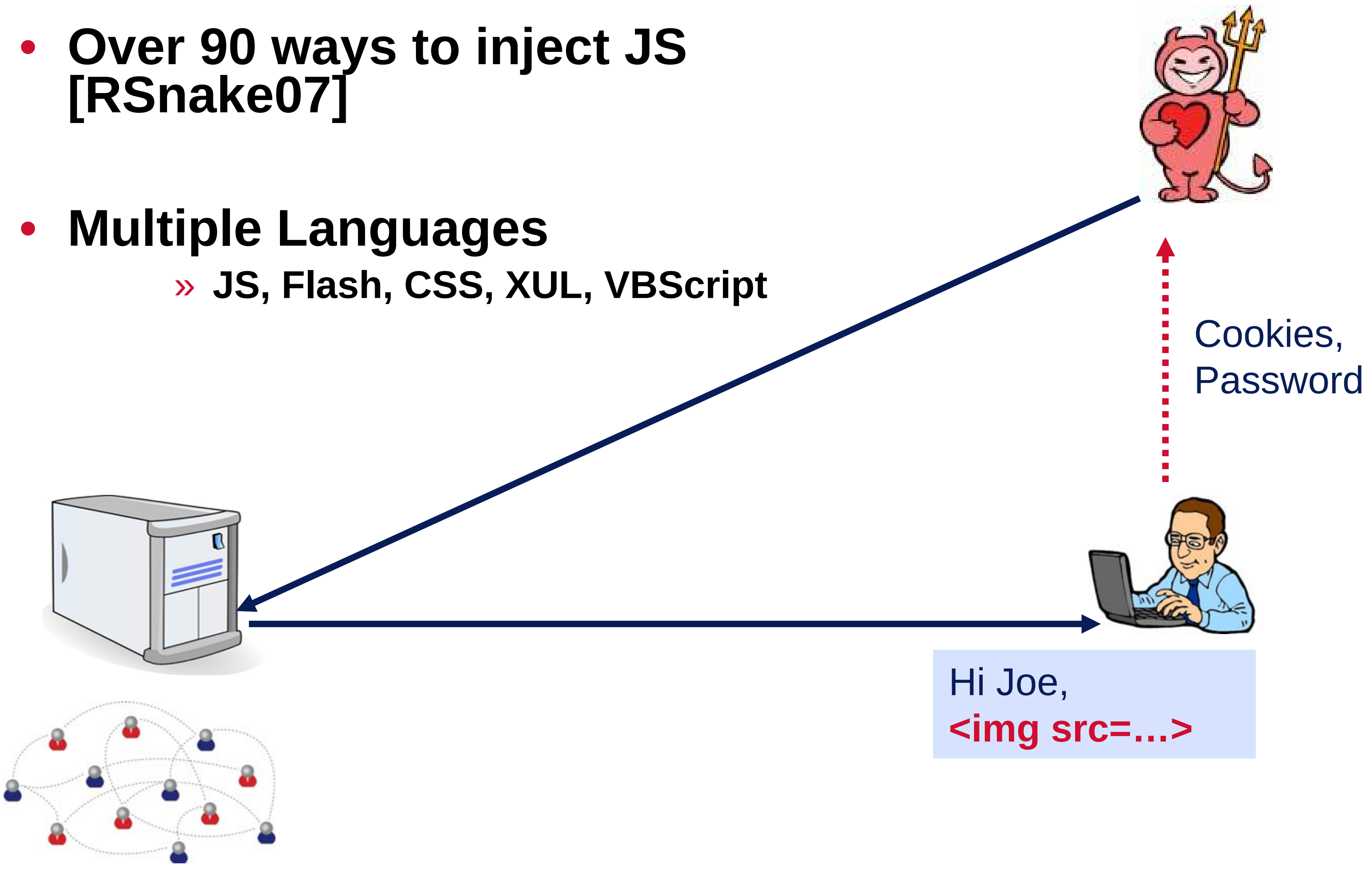


Policy: ALLOW
{a, a@href, img, img@src }

Hi Joe,
<img src=…>

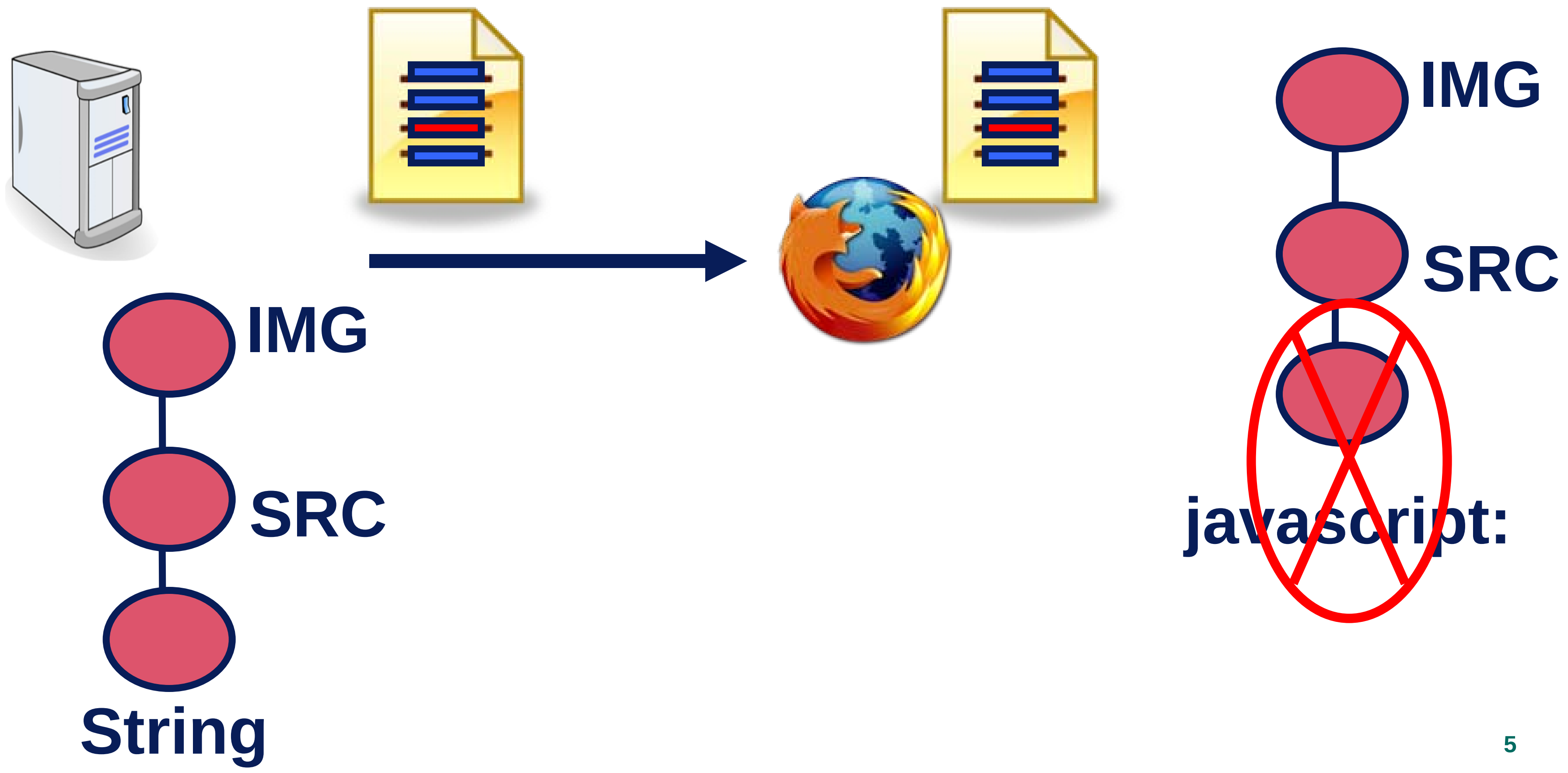
Limitations of Server-side Sanitization

- **Over 90 ways to inject JS [RSnake07]**
- **Multiple Languages**
  - » **JS, Flash, CSS, XUL, VBScript**

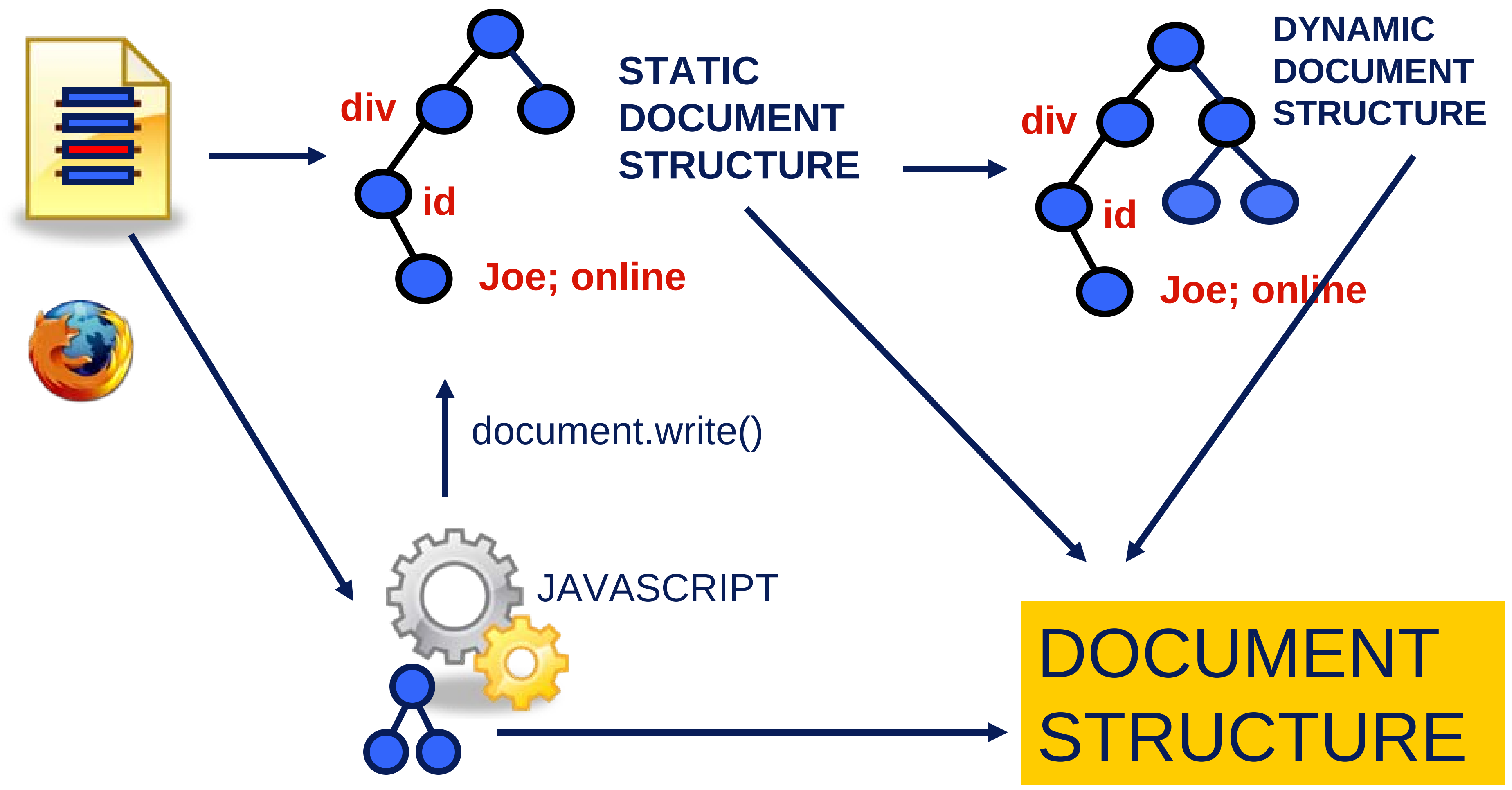


A Different Approach…

- **Previous defenses: XSS is a sanitization problem**
- **Our view: XSS is a document structure integrity problem**



Concept of Document Structure



Document Structure Integrity (DSI)

- **Definition:**
  - Given a server's policy P,
  - Restrict untrusted content to allowable syntactic elements
  - Policy in terms of client-side languages

- **Central idea for DSI enforcement**
  - Dynamic information flow tracking (server & browser)
  - Policy based parser-level confinement

- **Default policy: Only leaf nodes untrusted**

Talk Outline

- Power of DSI Defense: Examples
- Design Goals
- Architecture
- Implementation
- Evaluation
- Conclusion & Related Work

Talk Outline

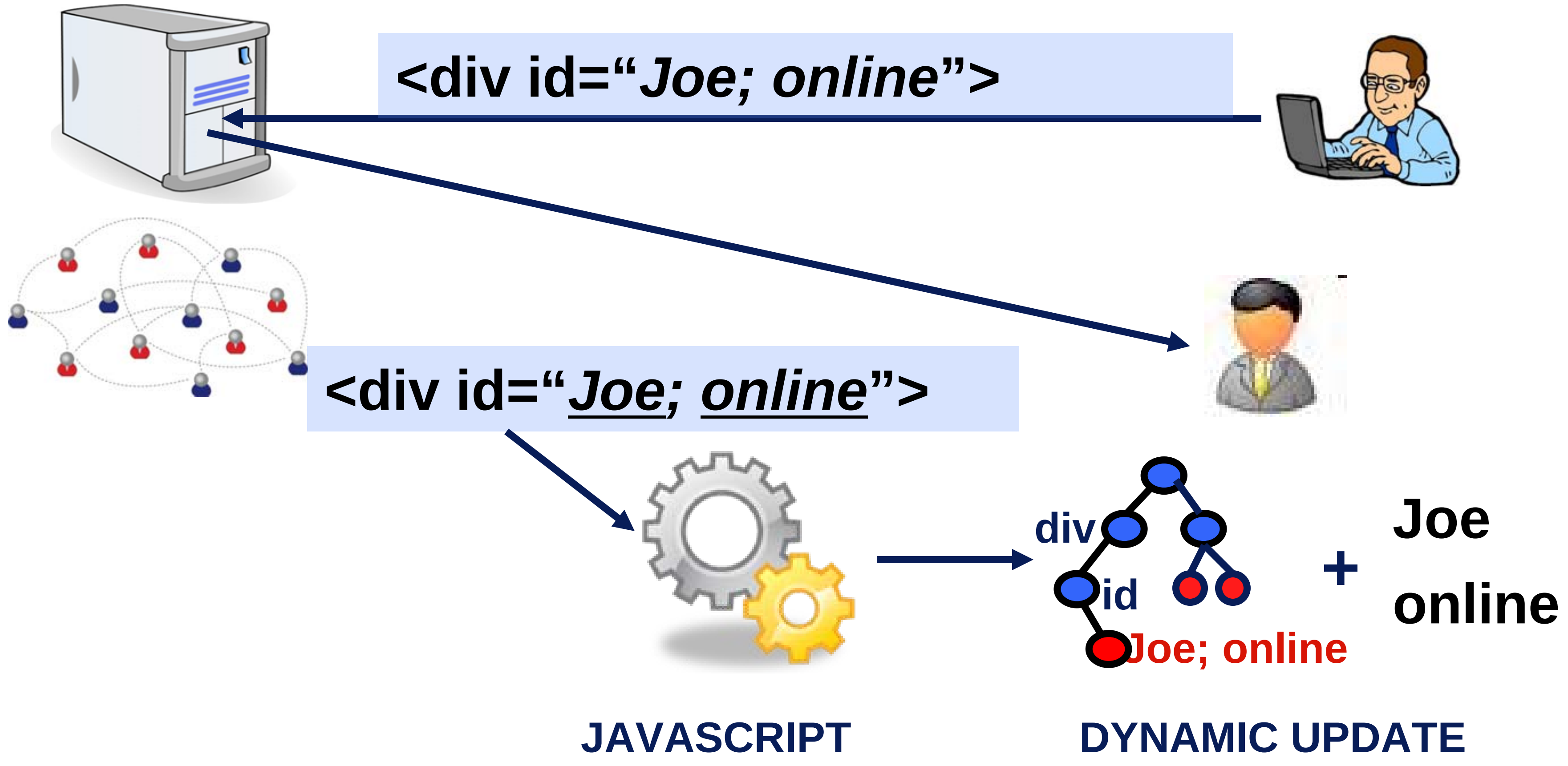
- **Power of DSI Defense: Examples**
- Design Goals
- Architecture
- Implementation
- Evaluation
- Conclusion & Related Work

DSI Defense: A Powerful Approach

- **DSI enforcement prevents**
  - **Not just cookie-theft**
    - » **Form injection for phishing [Netcraft08]**
    - » **Profile Worms [Samy05, Yammaner06]**
    - » **Web site defacement through XSS**
  - **“DOM-Based” XSS (Attacks on client-side languages)**
  - **Vulnerabilities due to browser-server inconsistency**

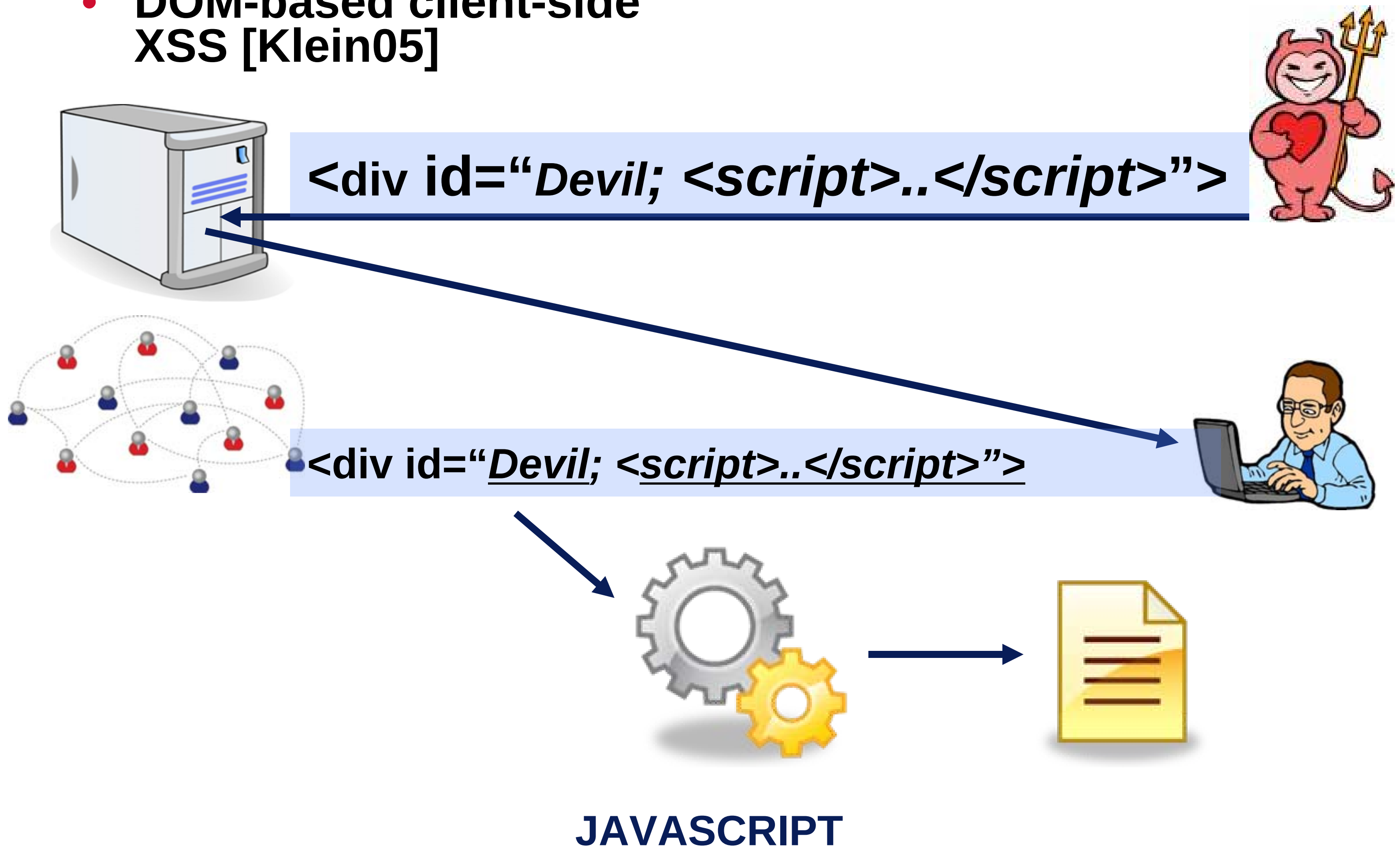
Example 1: DOM-Based XSS

- **DOM-based client-side XSS [Klein05]**



Example 1: DOM-Based XSS

- **DOM-based client-side XSS [Klein05]**

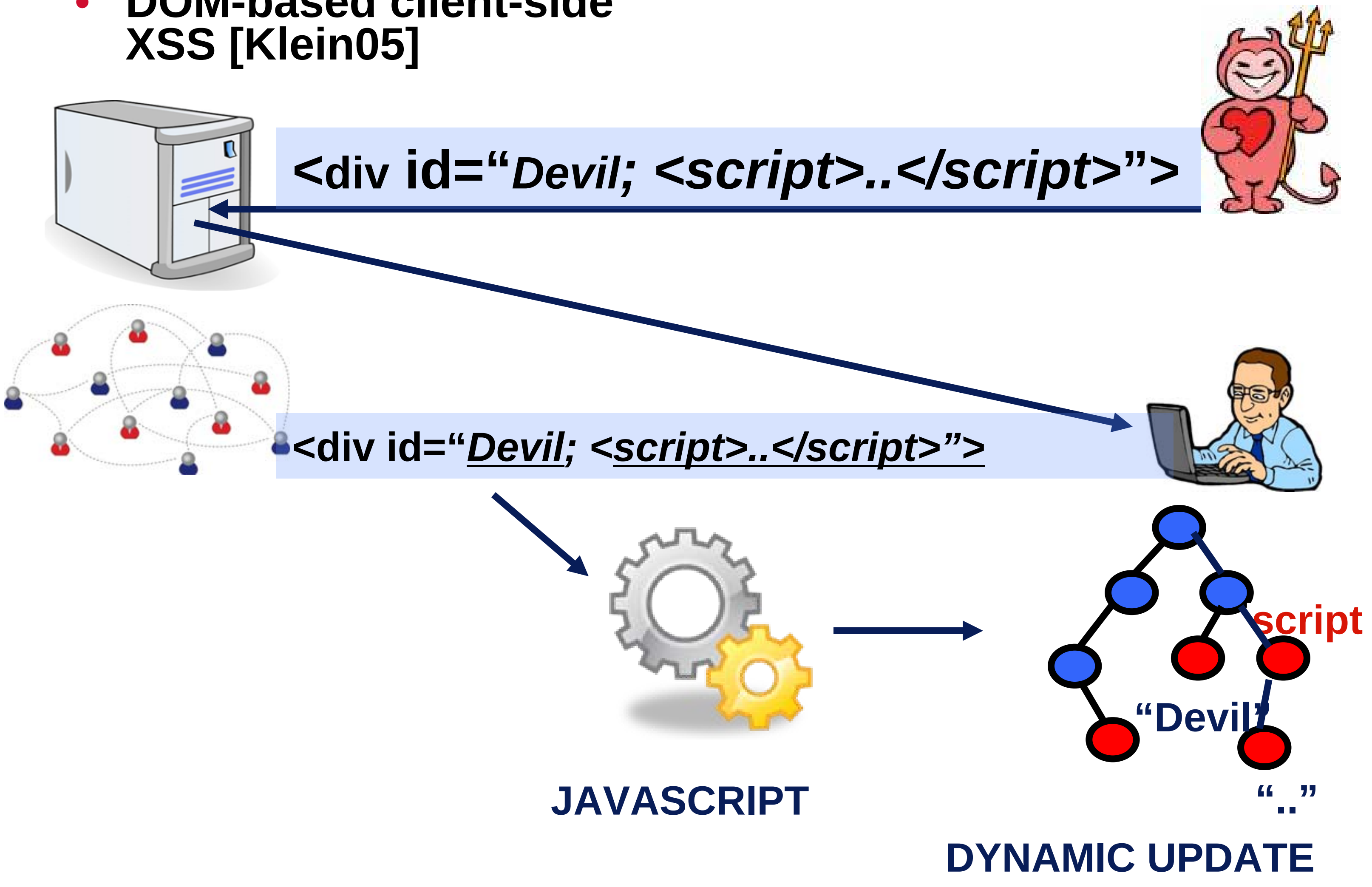


```
<div id="Devil; <script>..</script>">
```

```
<div id="Devil; <script>..</script>">
```

Example 1: DOM-Based XSS

- **DOM-based client-side XSS [Klein05]**

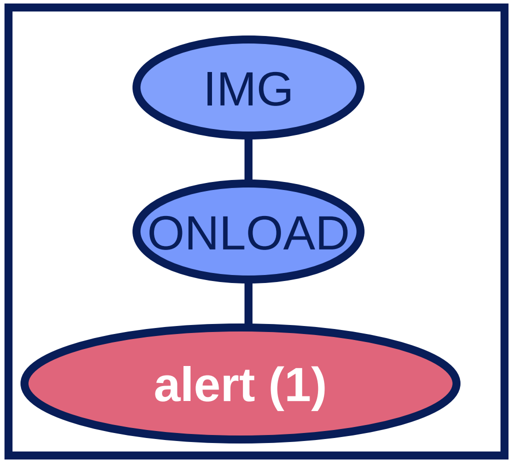


```
<div id="Devil; <script>..</script>">
```

```
<div id="Devil; <script>..</script>">
```

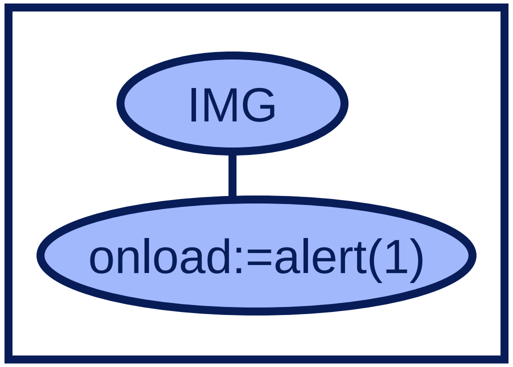
Example 2: Inconsistency Bugs

- **Browser-Server Inconsistency Bugs**



<img onload=alert(1)>

<img onload:=alert(1)>



<img onload:=alert(1)>

Assumed Parse Tree

Talk Outline

- Defense in Depth: Examples
- **Design Goals**
- Architecture
- Implementation
- Evaluation
- Conclusion & Related Work

Design Goals

- Clear separation between policy and mechanism
- No dependence on sanitization
- No changes to web application code
- Minimize false positives
- Minimizes impact to backwards compatibility
- Robustness
  - Address static & dynamic integrity attacks
  - Defeat adaptive adversaries

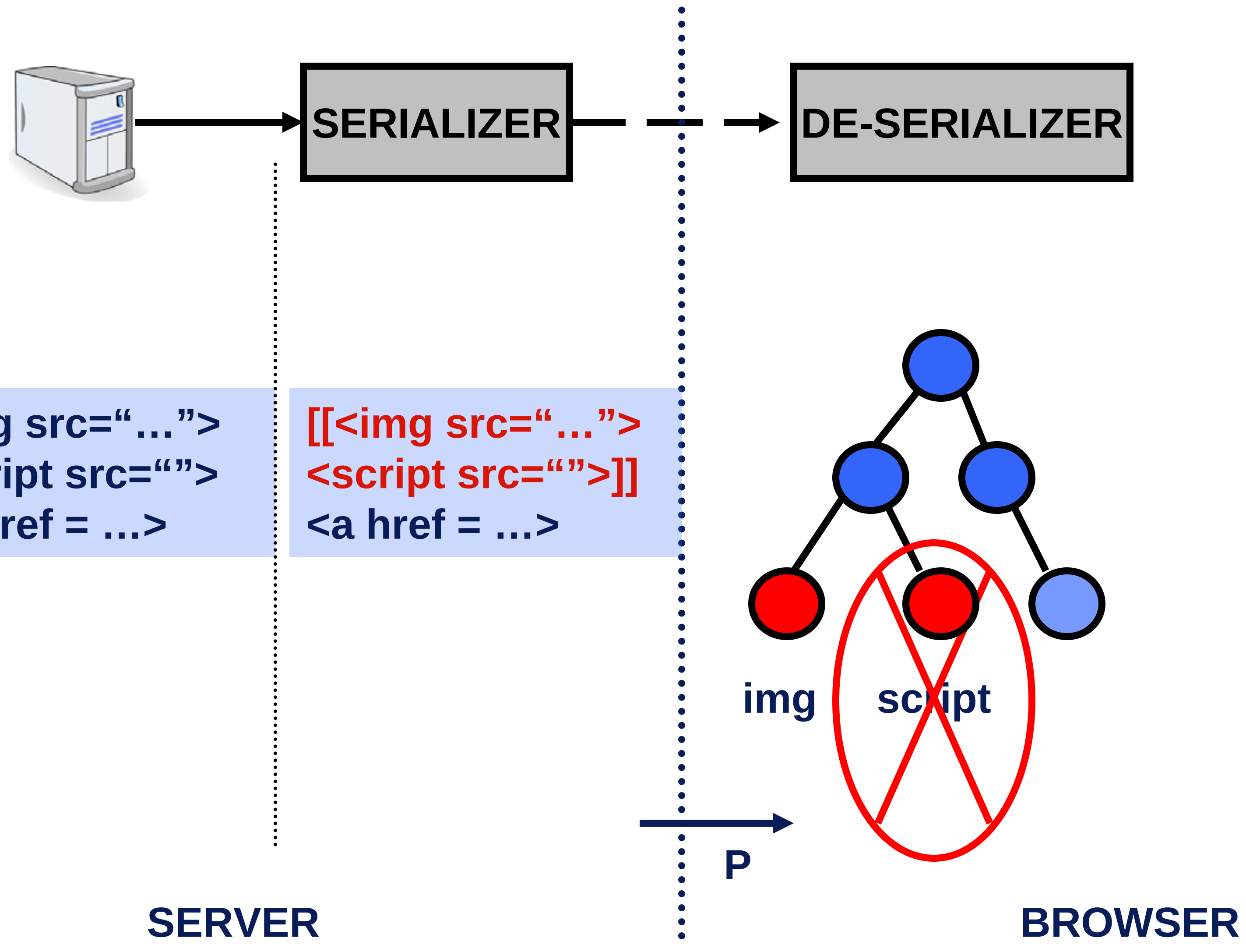
Mechanisms

- Client-server architecture
- Server
  - Step 1: Identify trust boundaries in HTML response
  - Step 2: Serialize
    - » Encoding data & trust boundaries in HTML
- Client
  - Step 3: De-serialize
    - » Initialize HTTP response page into static document structure
  - Step 4: Dynamic information flow tracking
    - » Modified semantics of client-side interpretation

Talk Outline

- Defense in Depth: Examples
- Design Goals
- **Architecture**
- Implementation
- Evaluation
- Conclusion & Related Work

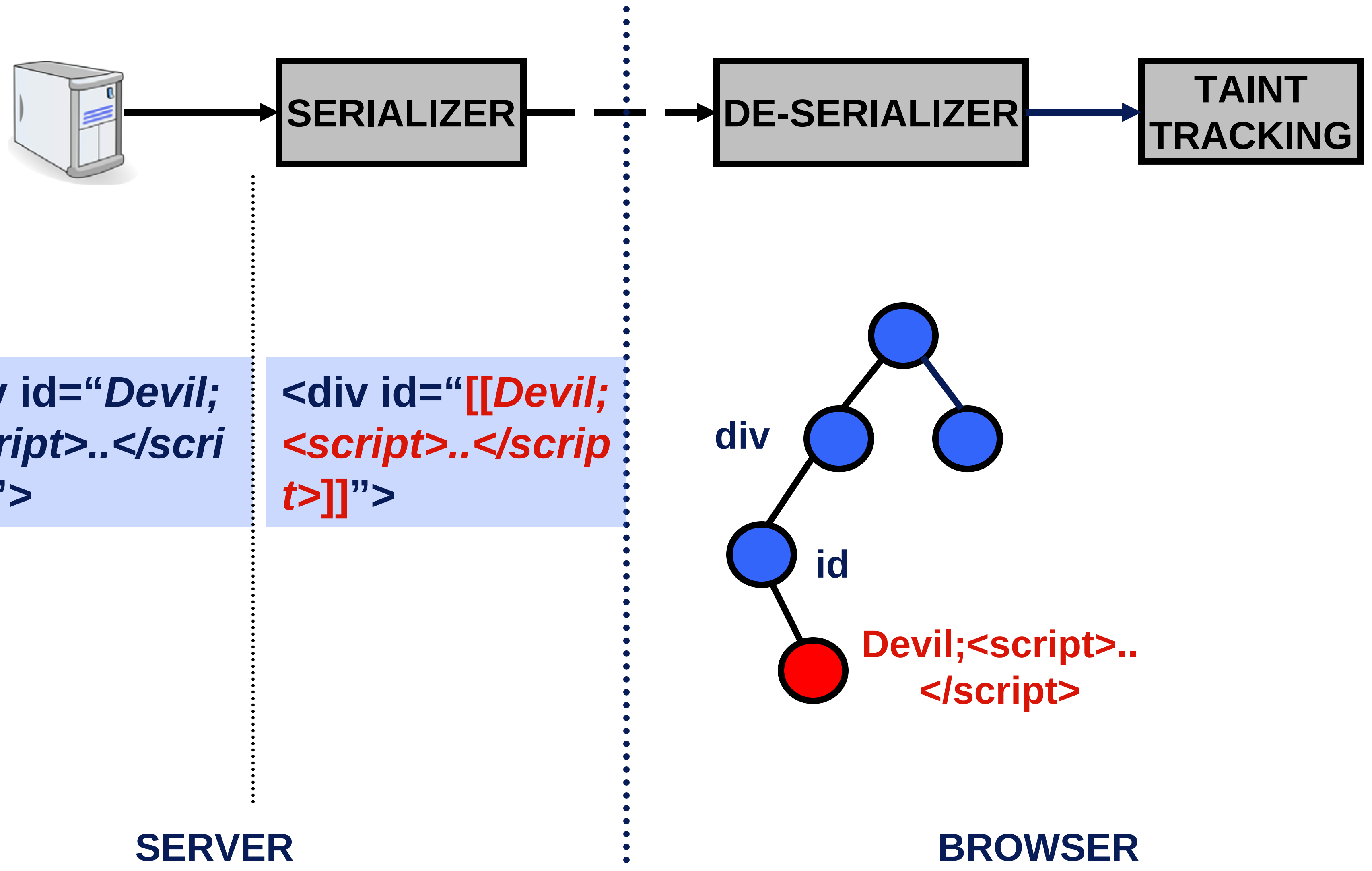
Approach Overview: Static DSI



```
<img src="…">
<script src="">
<a href = …>
```

```
[[<img src="…">
<script src="">]]
<a href = …>
```

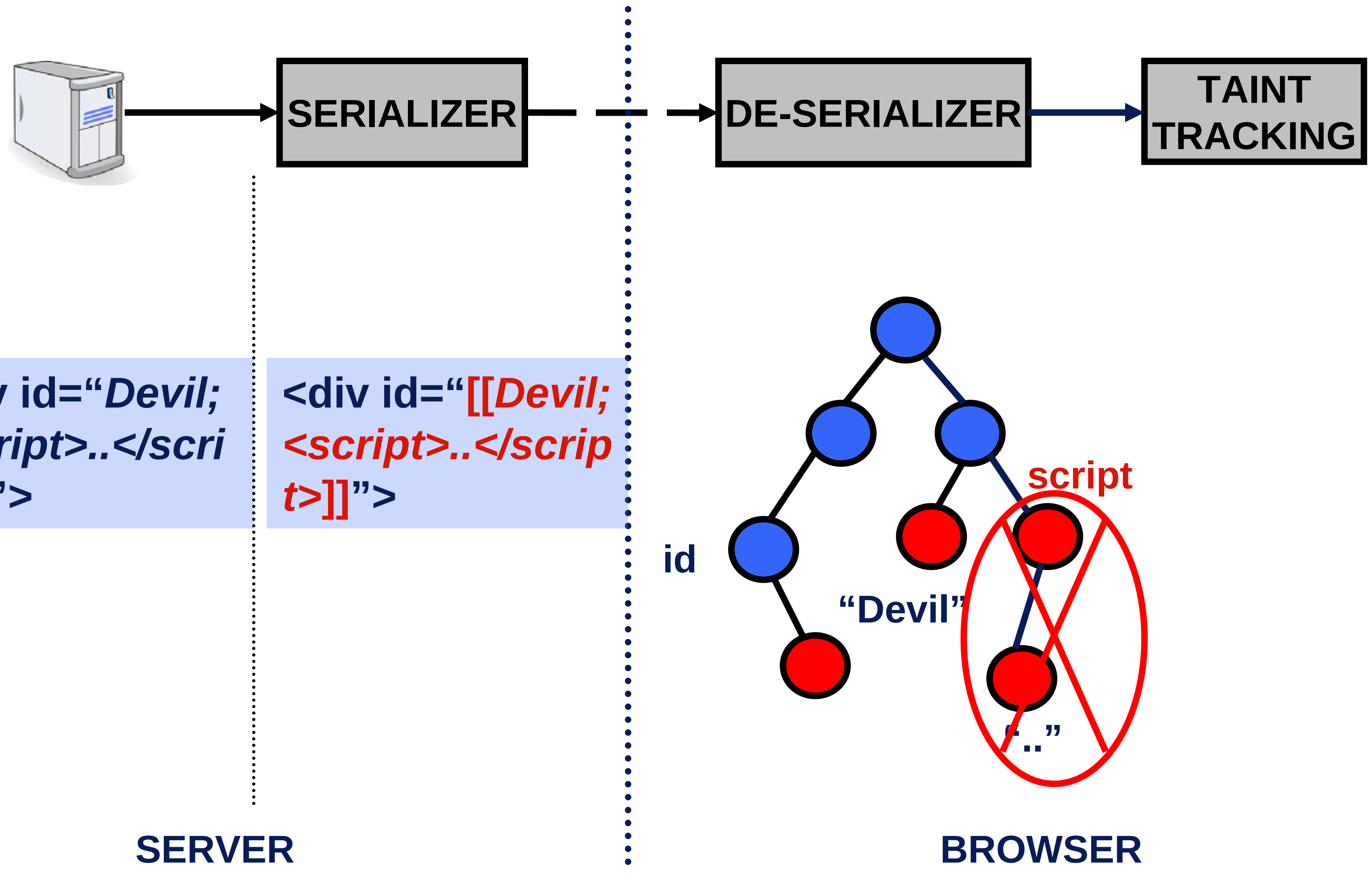
Approach Overview: Dynamic DSI



<div id=“*Devil; <script>..</script>*”>

<div id=“[[*Devil; <script>..</script>*]]”>

Approach Overview: Dynamic DSI (II)

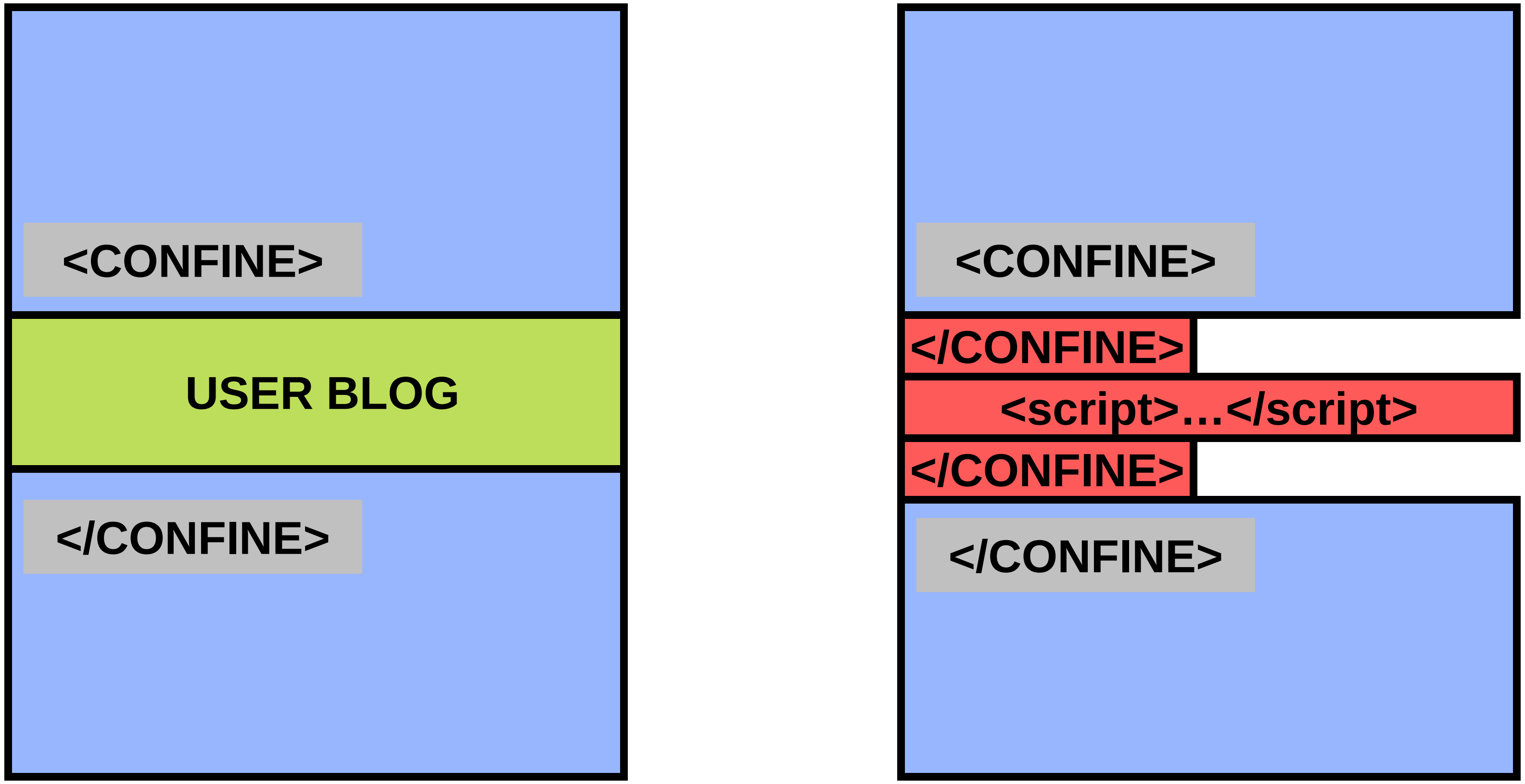


<div id="*Devil; <script>..</script>*">

<div id="[[*Devil; <script>..</script>*]]">

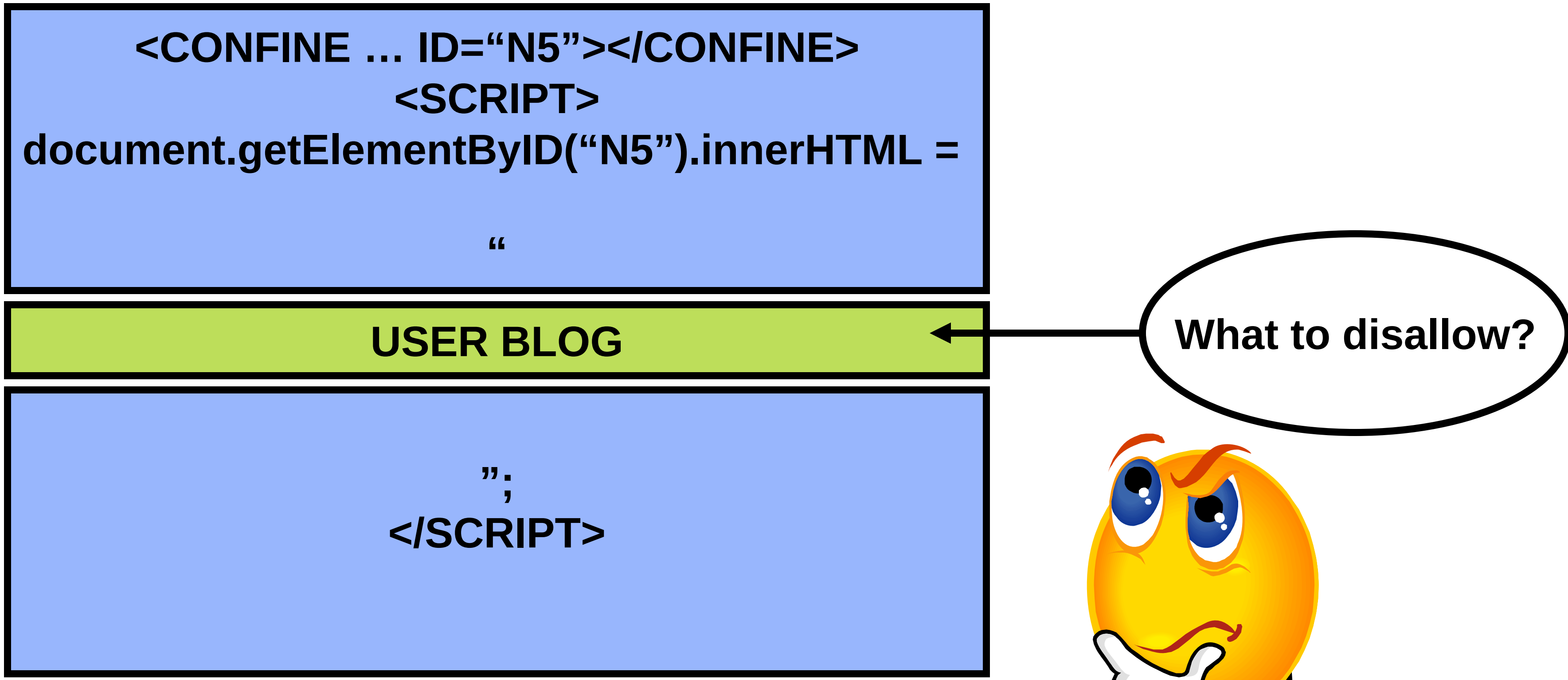
Serialization Design: Key Challenge

- **Safety against an adaptive adversary**



Serialization: Key Challenge

- Do not rely on sanitization



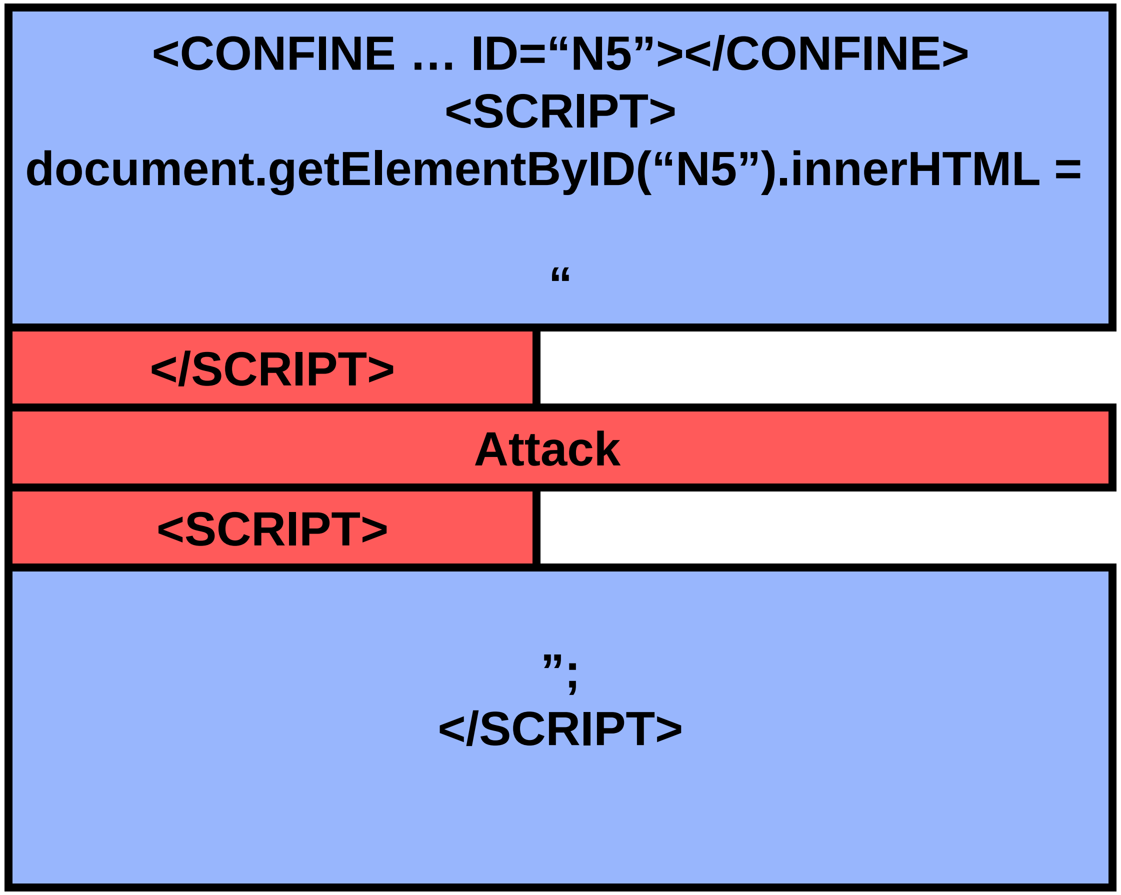
```
<CONFINE … ID="N5"></CONFINE>
<SCRIPT>
document.getElementByID("N5").innerHTML =

"
USER BLOG

";
</SCRIPT>
```

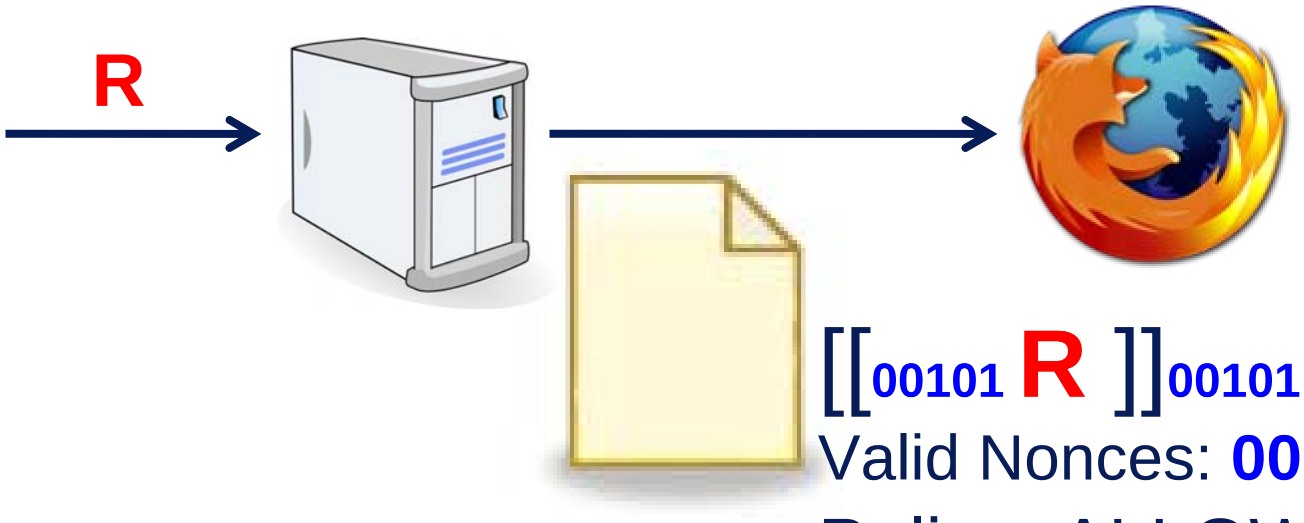
Serialization Design: Key Challenge

- **Attack on sanitization mechanism for JS strings**


<CONFINE … ID="N5"></CONFINE>
<SCRIPT>
document.getElementByID("N5").innerHTML =
"
</SCRIPT>
Attack
<SCRIPT>
";
</SCRIPT>

Markup Randomization

- **Markup Randomization**
  - **Mechanism independent of the policy**
  - **Does not depend on any sanitization**



[[$_{00101}$ R]]$_{00101}$

Valid Nonces: **00101**,**11010**,**01110**

Policy: ALLOW {a, a@aref ... }

Markup Randomization

- **Markup Randomization**
  - **Mechanism independent of the policy**
  - **Does not depend on any sanitization**

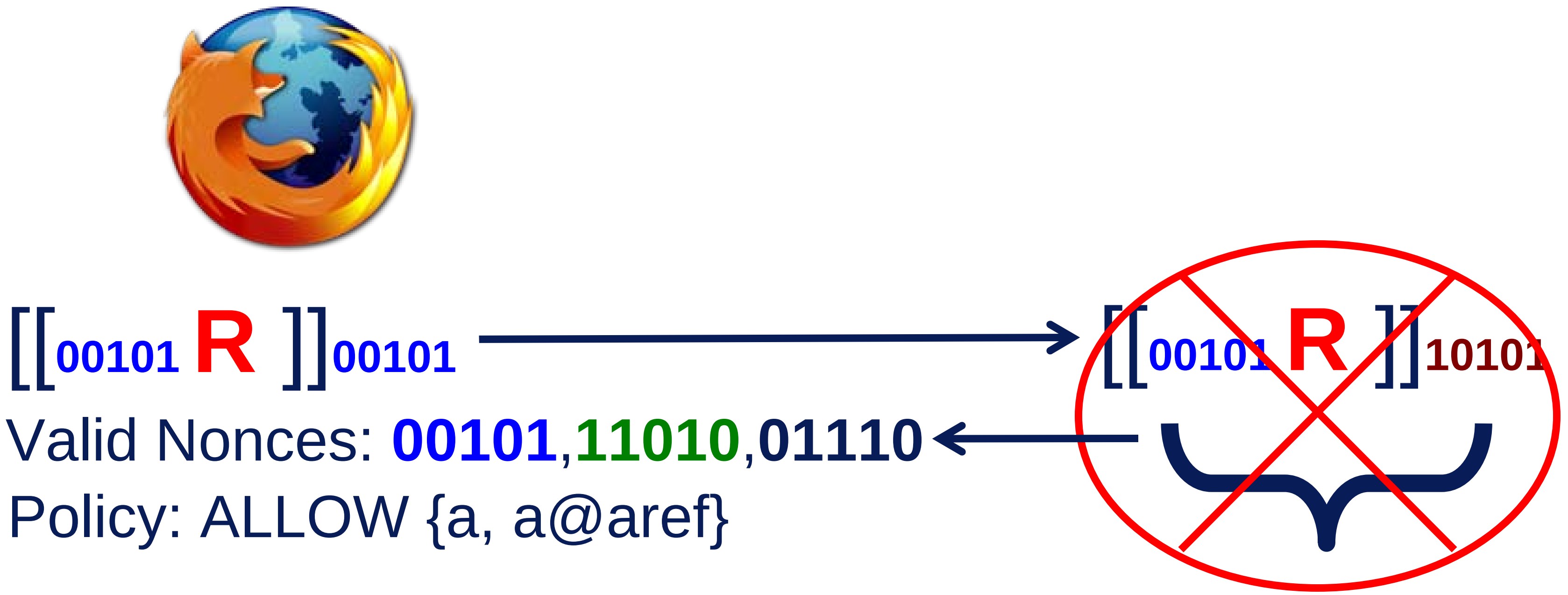
$$[[_{00101}\,\mathbf{R}\,]]_{00101} \longrightarrow \underbrace{[[_{00101}\,\mathbf{R}\,]]_{00101}}_{\textbf{OK!}}$$

Valid Nonces: **00101**,**11010**,**01110**

Policy: ALLOW {a, a@aref}

Markup Randomization

- **Markup Randomization**
  - **Mechanism independent of the policy**
  - **Does not depend on any sanitization**



Valid Nonces: **00101**,**11010**,**01110**

Policy: ALLOW {a, a@aref}

Browser-side Taint Tracking

- **Dynamic DSI**
- **Client Language Interpreters enhanced**
- **Ubiquitous tracking of untrusted data in the browser**

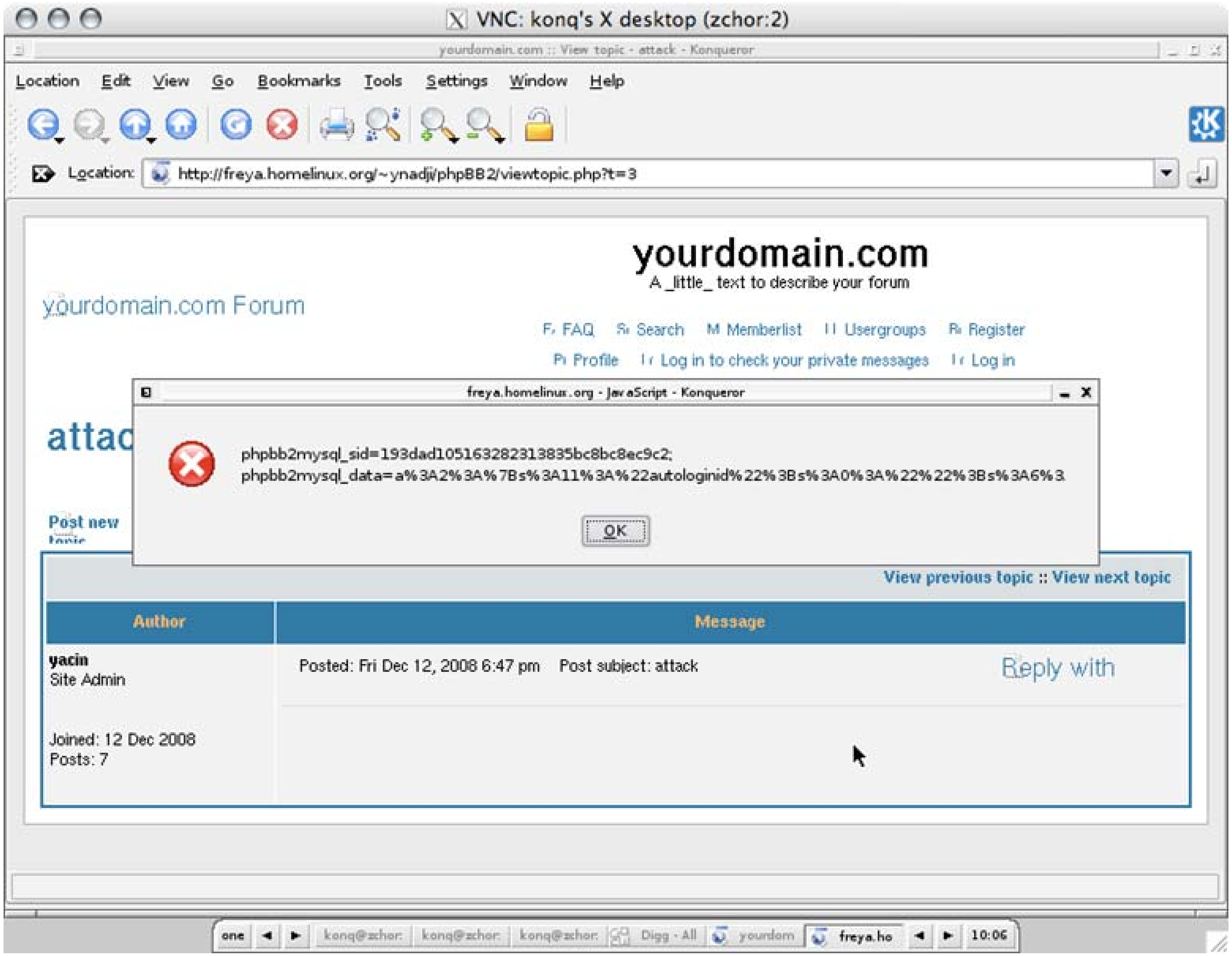
Talk Outline

- Advantages of DSI in Attack Coverage
- Design Goals
- Architecture
- **Implementation**
- Evaluation
- Conclusion & Related Work

Implementation

- **Full Prototype Implementation**
- **DSI-enable server**
  - **Utilized existing taint tracking in PHP [IBM07]**
- **DSI-compliant browser**
  - **Implemented in KDE Konqueror 3.5.9**
  - **Client side taint tracking in JS interpreter of KDE 3.5.9**

You are 0wned!


VNC: konq's X desktop (zchor:2)
yourdomain.com :: View topic - attack - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location: http://freya.homelinux.org/~ynadji/phpBB2/viewtopic.php?t=3
yourdomain.com
A _little_ text to describe your forum
yourdomain.com Forum
FAQ Search Memberlist Usergroups Register
Profile Log in to check your private messages Log in
freya.homelinux.org - JavaScript - Konqueror
OK
Post new topic
View previous topic :: View next topic
Author
Message
Reply with

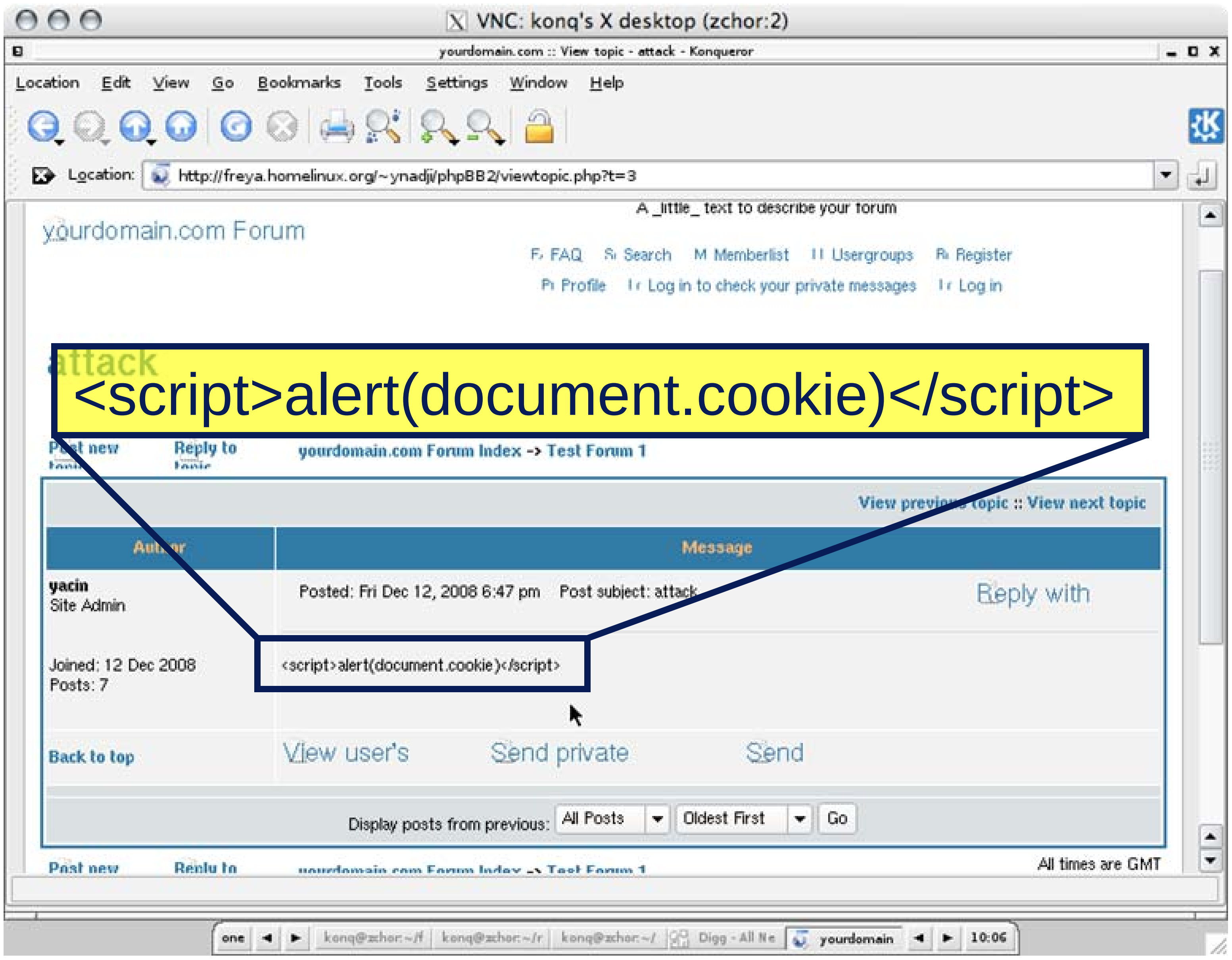
```
phpbb2mysql_sid=193dad105163282313835bc8bc8ec9c2;
phpbb2mysql_data=a%3A2%3A%7Bs%3A11%3A%22autologinid%22%3Bs%3A0%3A%22%22%3Bs%3A6%3
```

yacin
Site Admin

Joined: 12 Dec 2008
Posts: 7

Posted: Fri Dec 12, 2008 6:47 pm Post subject: attack

In a DSI-compliant Browser…


VNC: konq's X desktop (zchor:2)

yourdomain.com :: View topic - attack - Konqueror

Location: http://freya.homelinux.org/~ynadji/phpBB2/viewtopic.php?t=3

A _little_ text to describe your forum

yourdomain.com Forum

attack

<script>alert(document.cookie)</script>

yourdomain.com Forum Index -> Test Forum 1

View previous topic :: View next topic

Author	Message
yacin Site Admin Joined: 12 Dec 2008 Posts: 7	Posted: Fri Dec 12, 2008 6:47 pm Post subject: attack Reply with <script>alert(document.cookie)</script>
Back to top	View user's Send private Send

Display posts from previous: All Posts Oldest First Go

yourdomain.com Forum Index -> Test Forum 1

All times are GMT

Talk Outline

Evaluation: Attack Detection

- Stored XSS attacks

- Vulnerable phpBB forum application
- 25 public attack vectors [RSnake07]
- 30 benign posts

- Results
  - 100% attack prevention
  - No changes required to the application
  - No false positives

Evaluation: Real-World XSS Attacks

- **5,328 real-world vulnerabilities [xssed.com]**
- **500 most popular benign web sites [alexa.com]**
- **Default Policy:**
  - **Coerce untrusted data to leaf nodes**
- **Results**
  - **98.4% attack prevention**
  - **False Negatives:**
    - **Due to exact string matching in instrumentation**
  - **False Positives: 1%**
    - **Due to instrumentation for tainting (<title> on Slashdot)**

Evaluation: Performance

Browser Overhead	**1.8%**
Server overhead	**1-3%**
Static page size increase	**1.1%**

Related Work

- Client-server Approaches
  - » BEEP [Jim07]
  - » <jail> [Eich07]
  - » Hypertext Isolation [Louw08]
- Client-side approaches
  - » IE 8 Beta XSS Filter [IE8Blog]
  - » Client-side Firewalls [Kirda06]
  - » Sensitive Info. Flow Tracking [Vogt07]
- Server-side approaches
  - » Server-side taint-based defenses [Xu06, Nan07, Ngu05, Pie04]
  - » XSS-Guard [Bisht08]
  - » Program Analysis for XSS vulnerabilities [Balz08, Mar05, Mar08, Jov06, Hua04]

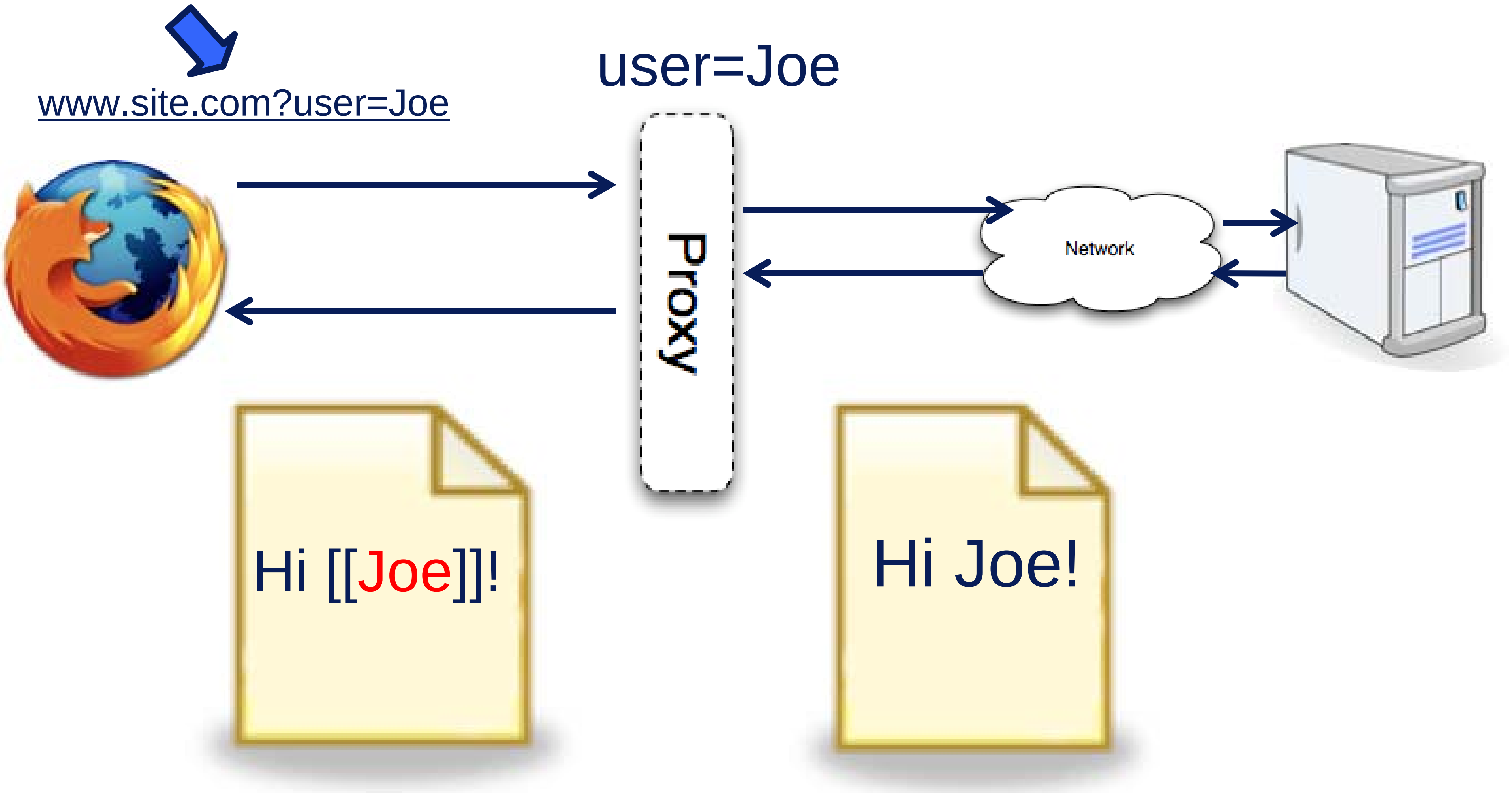
Conclusion

- DSI: A fundamental integrity property for web applications
- XSS as a DSI violation
- Multifaceted Approach
  - Clearly separates mechanism and policy
- Defeats adaptive adversaries
  - Markup randomization
- Evaluation on a large real-world dataset
  - Low performance overhead
  - No web application code changes
  - No false positives with configurable policies

Questions

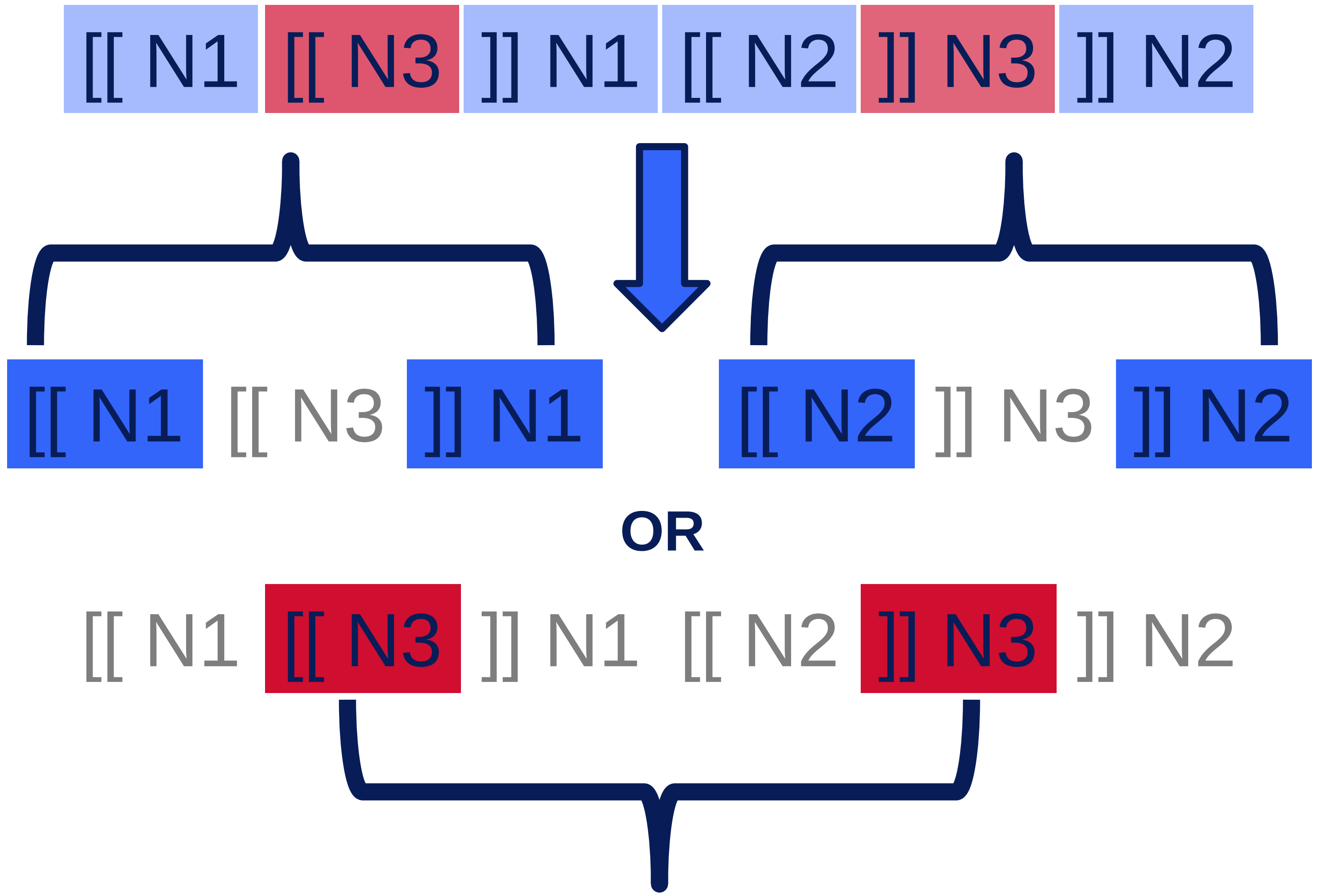
Thank you!

Client-Side Proxy



Markup Randomization: Adaptive Attacks

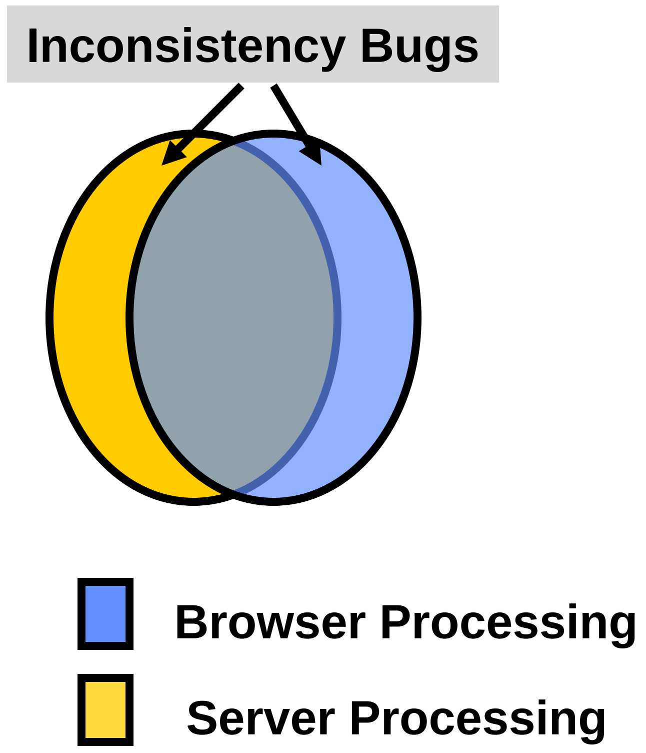
- **Multiple valid parse trees**



[[N1 [[N3]] N1 [[N2]] N3]] N2
[[N1 [[N3]] N1
[[N2]] N3]] N2
OR
[[N1 [[N3]] N1 [[N2]] N3]] N2

Attack Coverage (II): Inconsistency Bugs

- **Browser-Server Inconsistency Bugs**


Inconsistency Bugs
Browser Processing
Server Processing