Chrome Extension Introduction

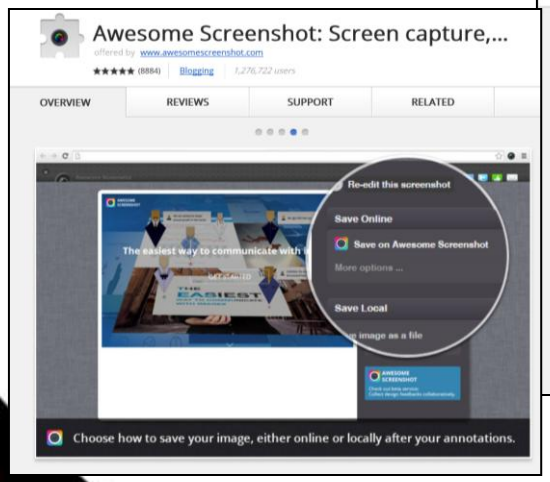
Presenter: Jienan Liu

Network, Intelligence & security Lab

What is Chrome Extension

- **Extension**
  - Small software programs that can modify and enhance the functionality of the Chrome browser.
  - Written with web technologies, such as HTML, Javascript, and CSS.

Screenshot

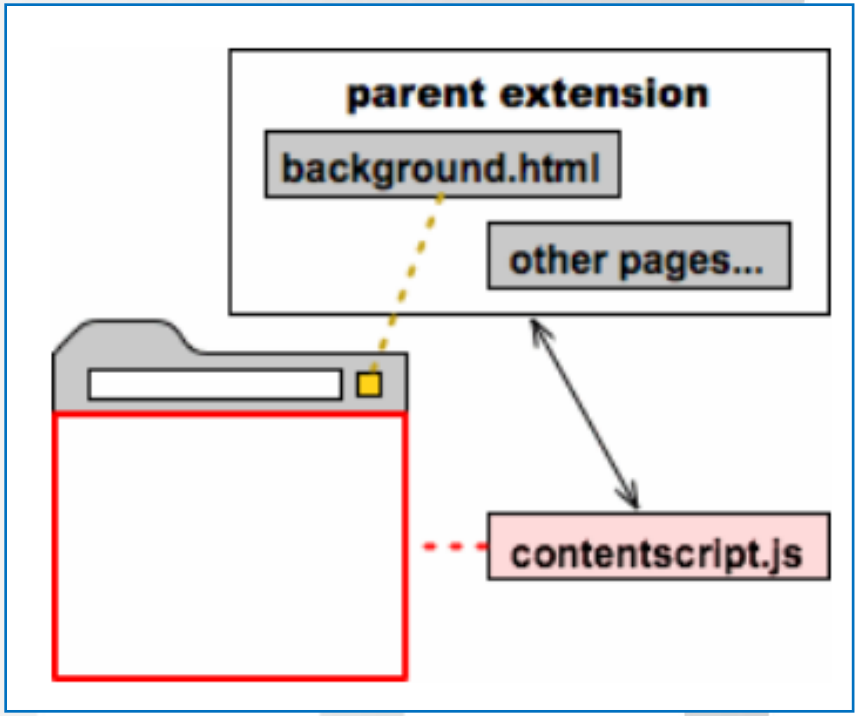


Ad Block

Pwd Protection

Chrome Extension Architecture

- **Components**
  - **Background page**
    - Holds main logic
    - Can include Javascript code
  - **UI pages**
    - Ordinary HTML pages
    - display the extension's UI
  - **Content script**
    - Interact with user web page
    - Javascript that executes in user's page
    - execute in a special environment



Chrome Extension Files

- Each extension has the following files:
  - A **manifest file**
  - One or more **HTML files** (unless the extension is a theme)
  - *Optional:* One or more **JavaScript files**
  - *Optional:* Any other files your extension needs—for example, image files
- Put all these files in one single folder while developing
- The contents of the folder are packaged into a special ZIP file when you distribute your extension

Manifest File

- Every extension has a JSON-formatted manifest file, named manifest.json
- Give information about the extension
  - Important files / capabilities that the extension may use
  - Permissions that extension needed

```
{
  "name": "My Extension",
  "version": "2.1",
  "description": "Gets information from Google.",
  "icons": { "128": "icon_128.png" },
  "background": {
    "persistent": false,
    "scripts": ["bg.js"]
  },
  "permissions": ["http://*.google.com/", "https://*.google.com/"],
  "browser_action": {
    "default_title": "",
    "default_icon": "icon_19.png",
    "default_popup": "popup.html"
  }
}
```

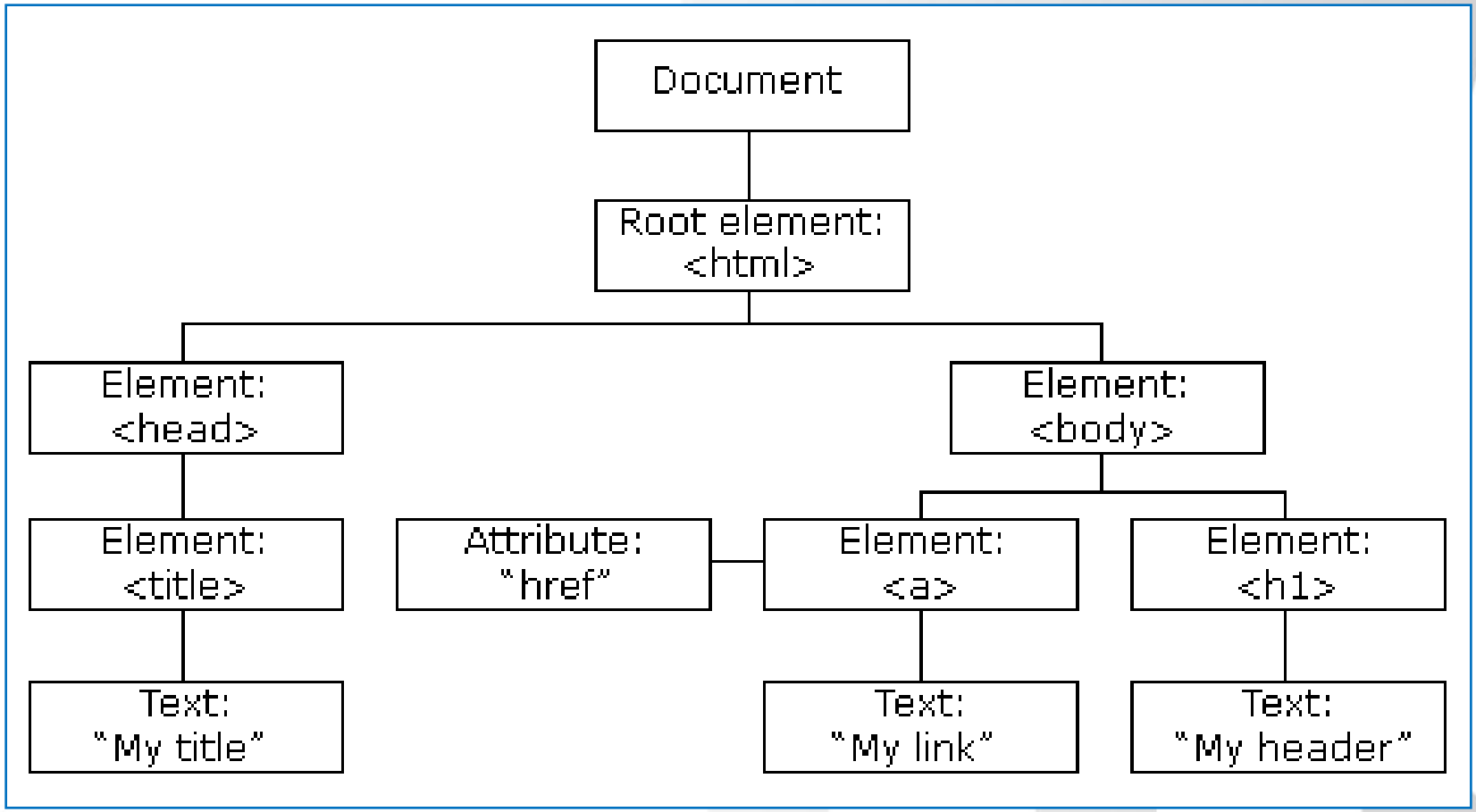
Example-1

- Chrome Extension Architecture

Content Scripts-1

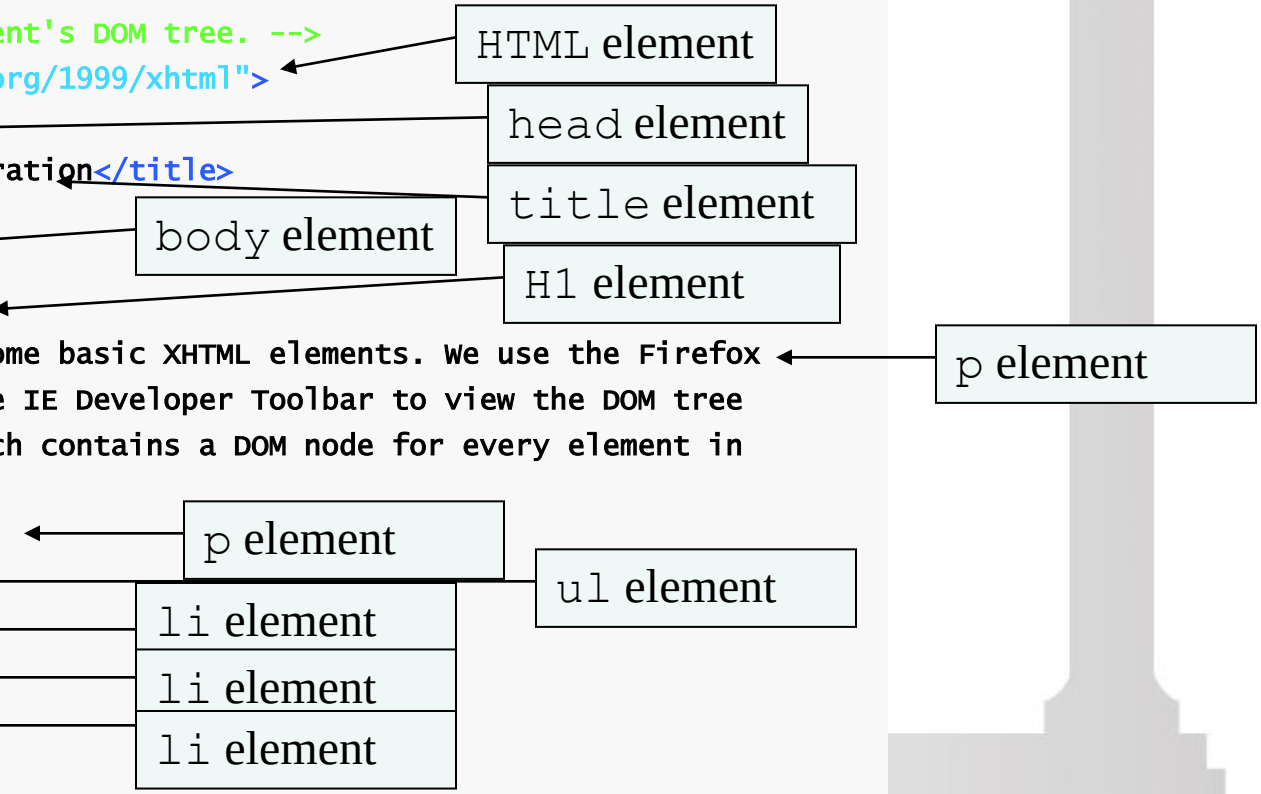
- Javascript files that run in the context of web pages
- Can read and modify Document Object Model (DOM) of the loaded pages
  - Provides a structured representation of the document
  - Defines a way that the structure can be accessed from programs
  - The Document Object Model gives you access to all the elements on a web page. Using JavaScript, you can create, modify and remove elements in the page dynamically.
  - DOM components form a tree of nodes
    - Relationship: parent node – children nodes
    - `document` is the root node
  - Attributes of elements are accessible as text

DOM Tree


Document
Root element:
<html>
Element:
<head>
Element:
<body>
Element:
<title>
Attribute:
"href"
Element:
<a>
Element:
<h1>
Text:
"My title"
Text:
"My link"
Text:
"My header"

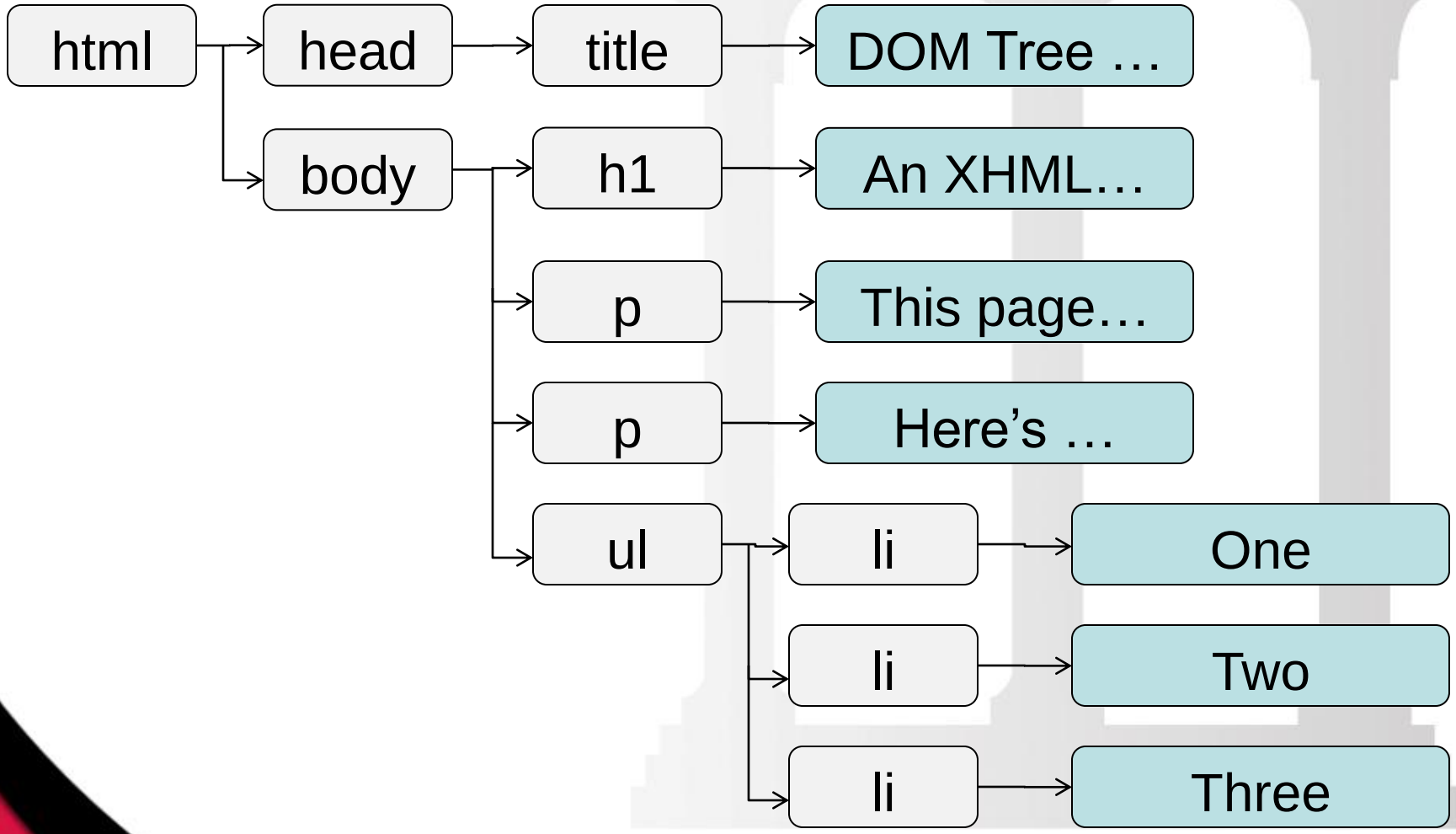
Demonstration of a document's DOM tree

```
1  <?xml version = "1.0" encoding = "utf-8"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. 12.1: domtree.html -->
6  <!-- Demonstration of a document's DOM tree. -->
7  <html xmlns = "http://www.w3.org/1999/xhtml">
8     <head>
9        <title>DOM Tree Demonstration</title>
10    </head>
11    <body>
12       <h1>An XHTML Page</h1>
13       <p>This page contains some basic XHTML elements. We use the Firefox
14          DOM Inspector and the IE Developer Toolbar to view the DOM tree
15          of the document, which contains a DOM node for every element in
16          the document.</p>
17       <p>Here's a list:</p>
18       <ul>
19          <li>One</li>
20          <li>Two</li>
21          <li>Three</li>
22       </ul>
23    </body>
24 </html>
```



Example-2

DOM Tree Demonstration



Core Interfaces in DOM

- document.getElementById(id)
- document.getElementsByTagName(name)
- document.createElement(name)
- parentNode.appendChild(node)
- element.innerHTML
- element.setAttribute()
- element.getAttribute()
- element.addEventListener()
- window.onload()

Example-3

- DOM object operation

Content Scripts-2

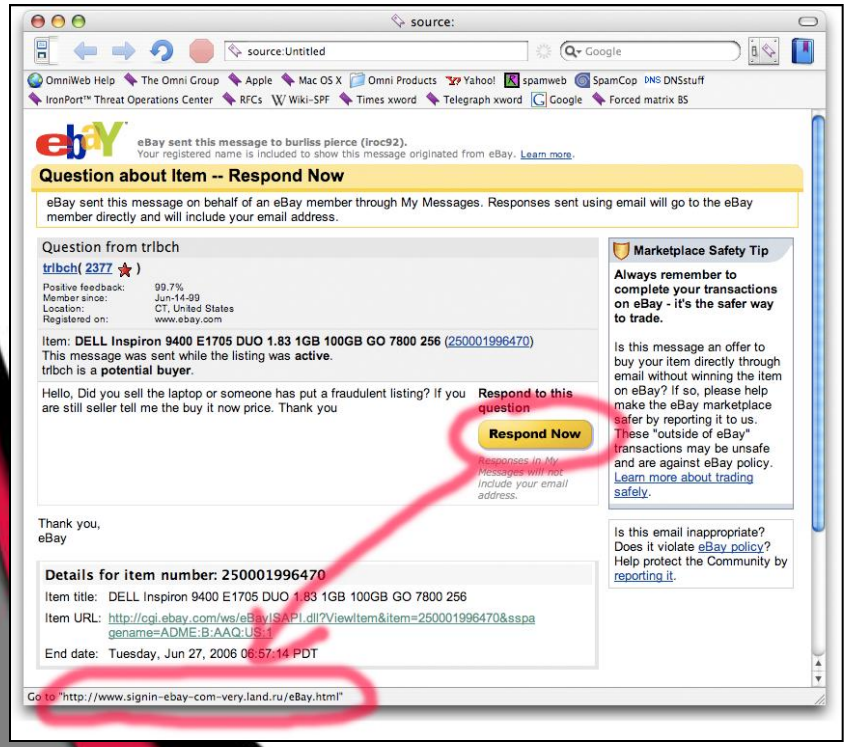
- Execute in a special environment called isolated world
  - Have access to the DOM of hosting page
  - No access to variables/functions created by the page
- Restricted to use limited Chrome.* APIs
- Use message passing to communicate with the rest of the extension.

Example-4

- Extension demonstration
  - Content script
  - Backgroud script

Motivation

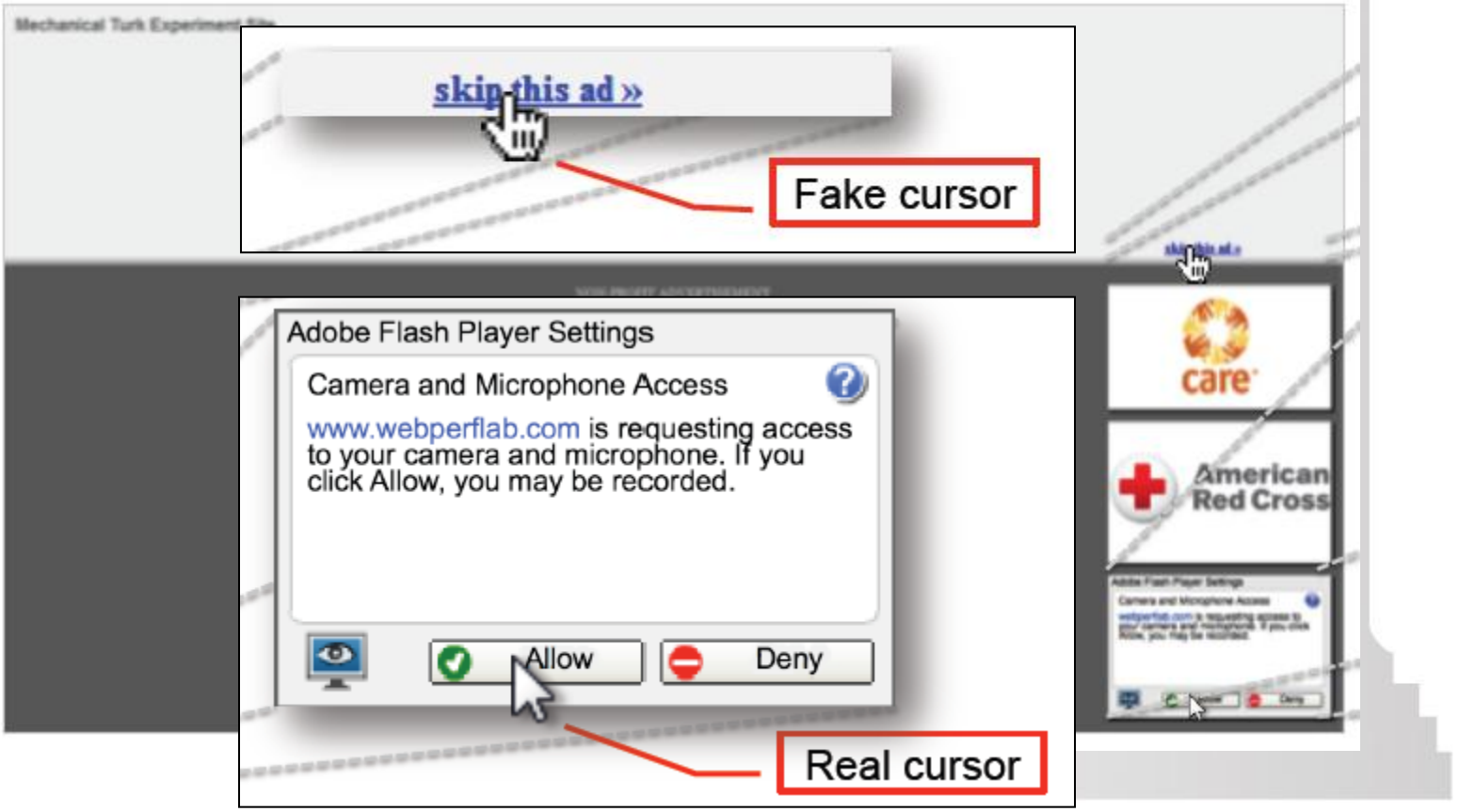
- Background
  - Threat from malicious web sites that trick users
  - Forensic analysis tools for web attacks, like phising, clickjacking, are not sufficient



Clickjacking Attack

Cursor Spoofing Attack

- Hide Flash webcam permission dialog inside Ads, and abuse pointer integrity



Our Goal

- High Level Idea
  - Take snapshot of Dom tree and Screen area for Chrome browser
  - Perform snapshot taking only for specific events, eg. mouse click, keystroke
  - Hook javascript event handler
- Feasibility
  - Event-driven javascript
  - DOM modifications by javascript code
- Solution
  - Implement our idea with Chrome extension

Thanks !

Q & A