Chrome Extension Security Architecture

Presenter: Jienan Liu

Network, Intelligence & security Lab

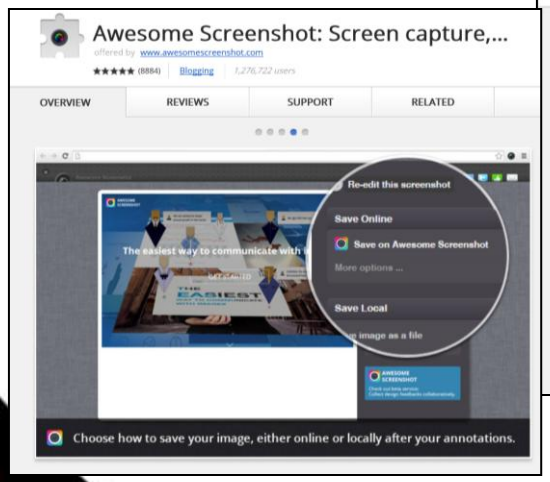
outline

- Chrome extension introduction
- Threats towards extension
- Chrome extension’s security architecture

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

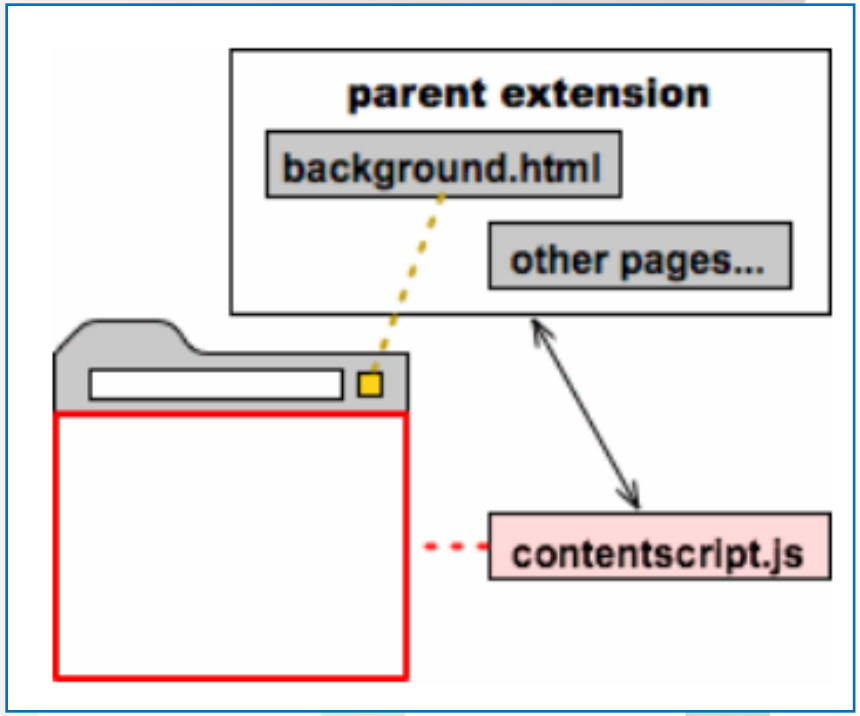


Ad Block

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 is executed in user's page
    - execute in a special environment



Chrome Extension Files

- One 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

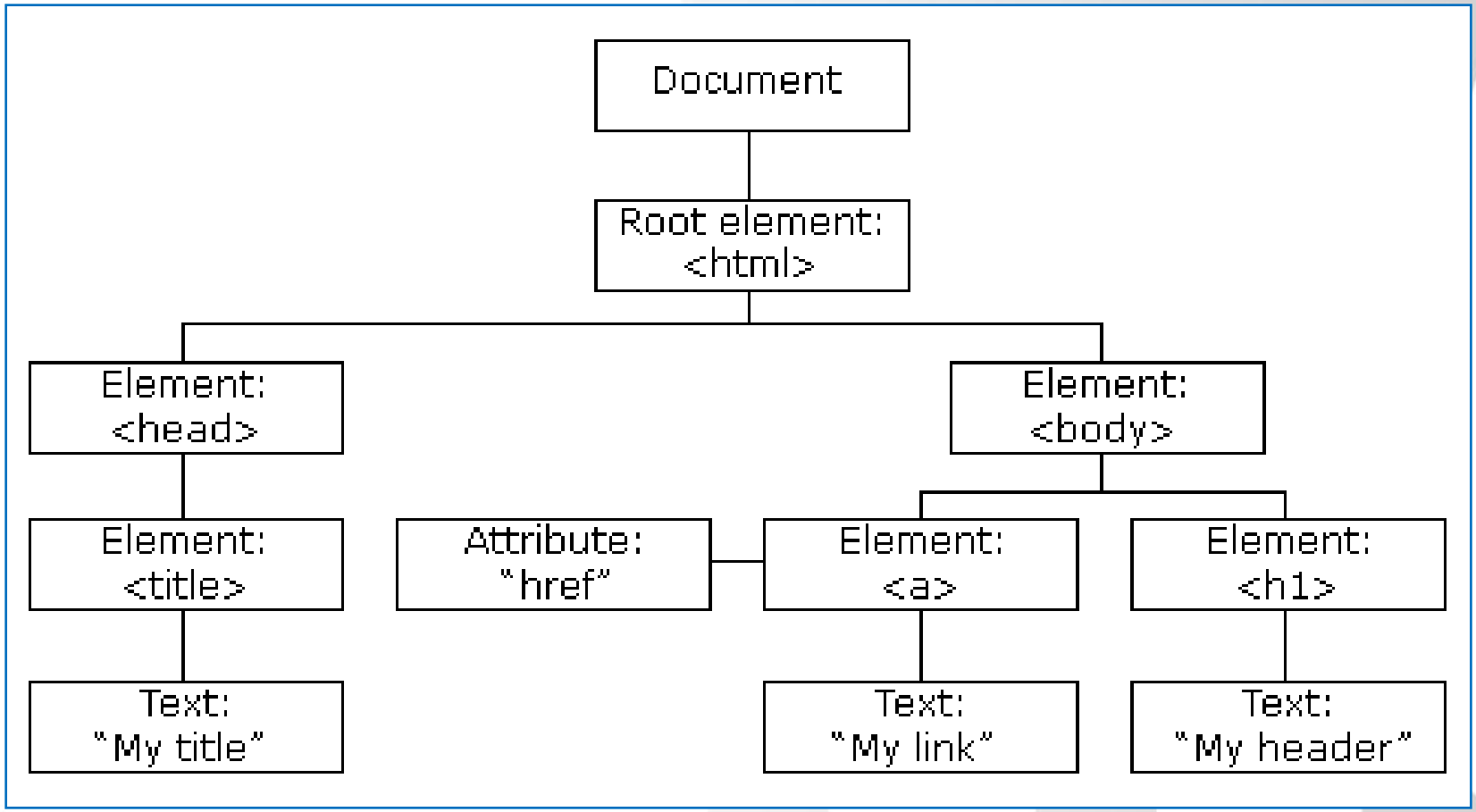
- Every extension has a JSON-formatted manifest file, named manifest.json
- Give information about the extension
  - Important files / capabilities that the extension may use
  - Permissions that extension needed

```
{
  "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"
  }
}
```

Content Scripts

- Javascript files that run in the context of web pages
- Can read and modify Document Object Model (DOM) of the loaded pages
  - What is DOM?
    - 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
      - `document` is the root node

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"

outline

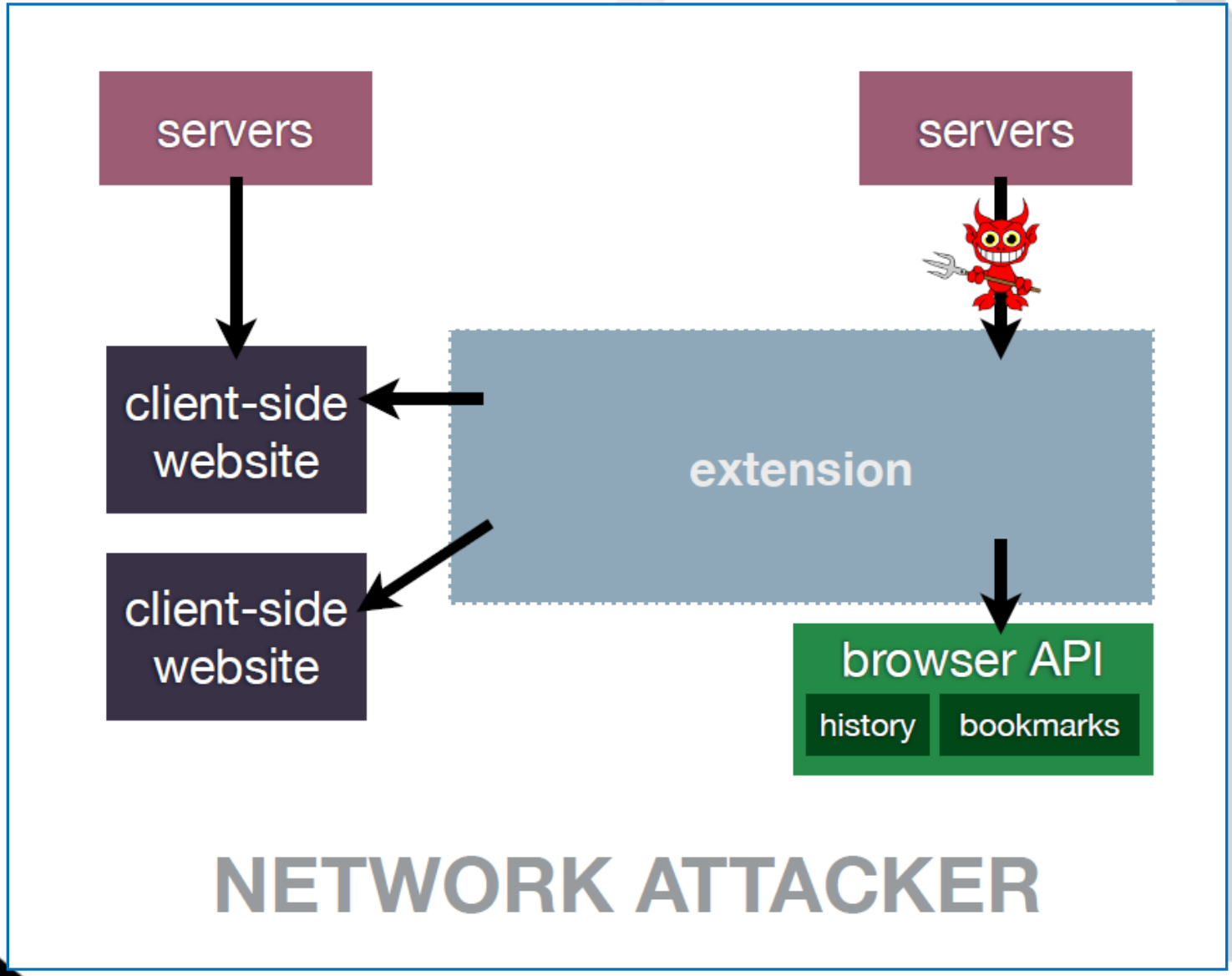
- Chrome extension introduction
- Threats towards extension
- Chrome extension security architecture

Extension security issues

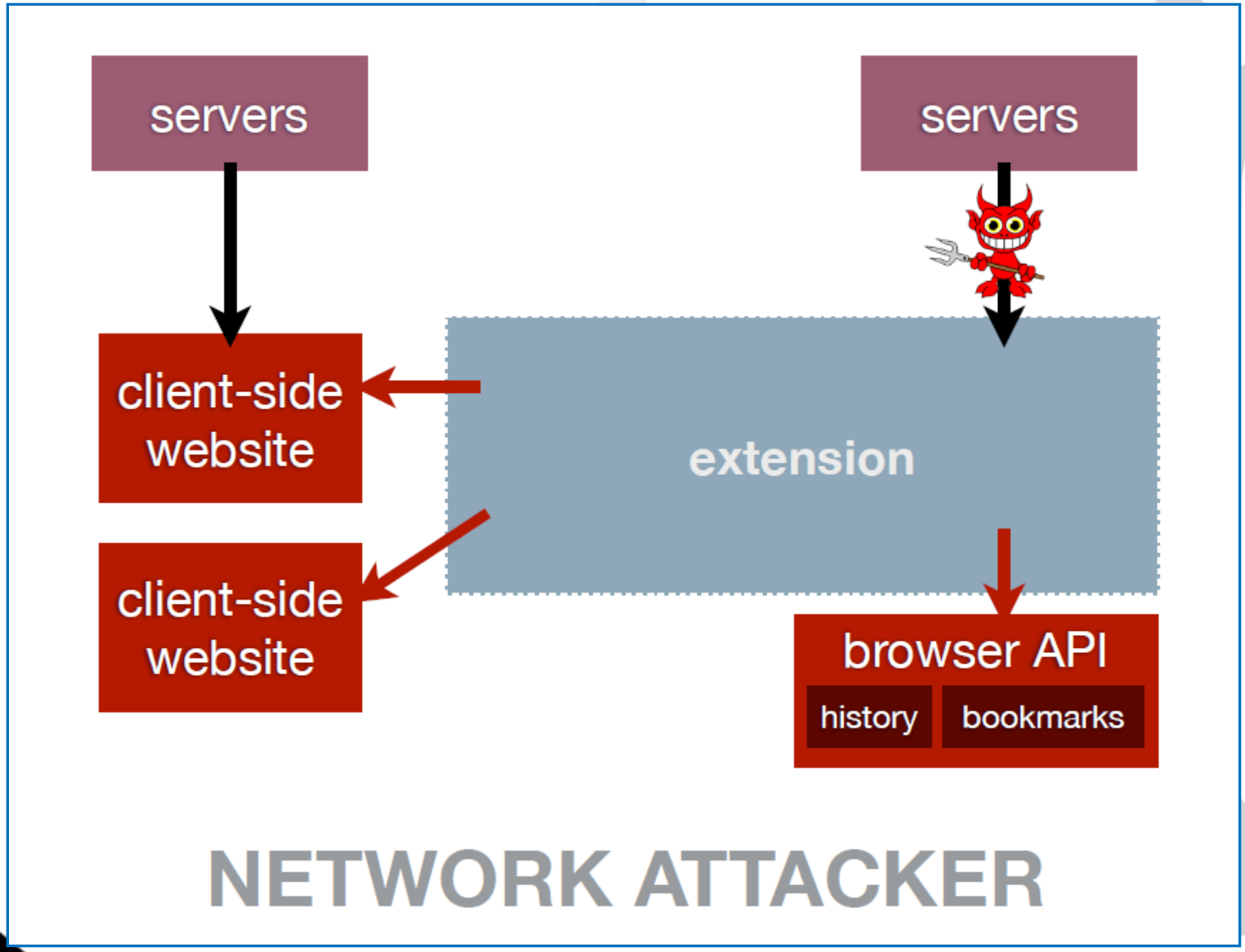
- Why extension could introduce vulnerabilities:
  - can read and manipulate content from websites, make unfettered network requests, and access browser user data like bookmarks and geolocation.
  - In the hands of a web or network attacker, these privileges can be abused to collect users' private information and authentication credentials.

- How extensions introduce vulnerabilities:
  - primarily written in JavaScript and HTML, and JavaScript provides several methods for converting strings to code, such as eval. If used improperly, these methods can introduce code injection vulnerabilities that compromise the extension.
  - Data can also execute if it is written to a page as HTML instead of as text, e.g., through the use of document.write or document.body.innerHTML. Extension developers may be not careful to avoid passing untrusted data to these execution sinks.

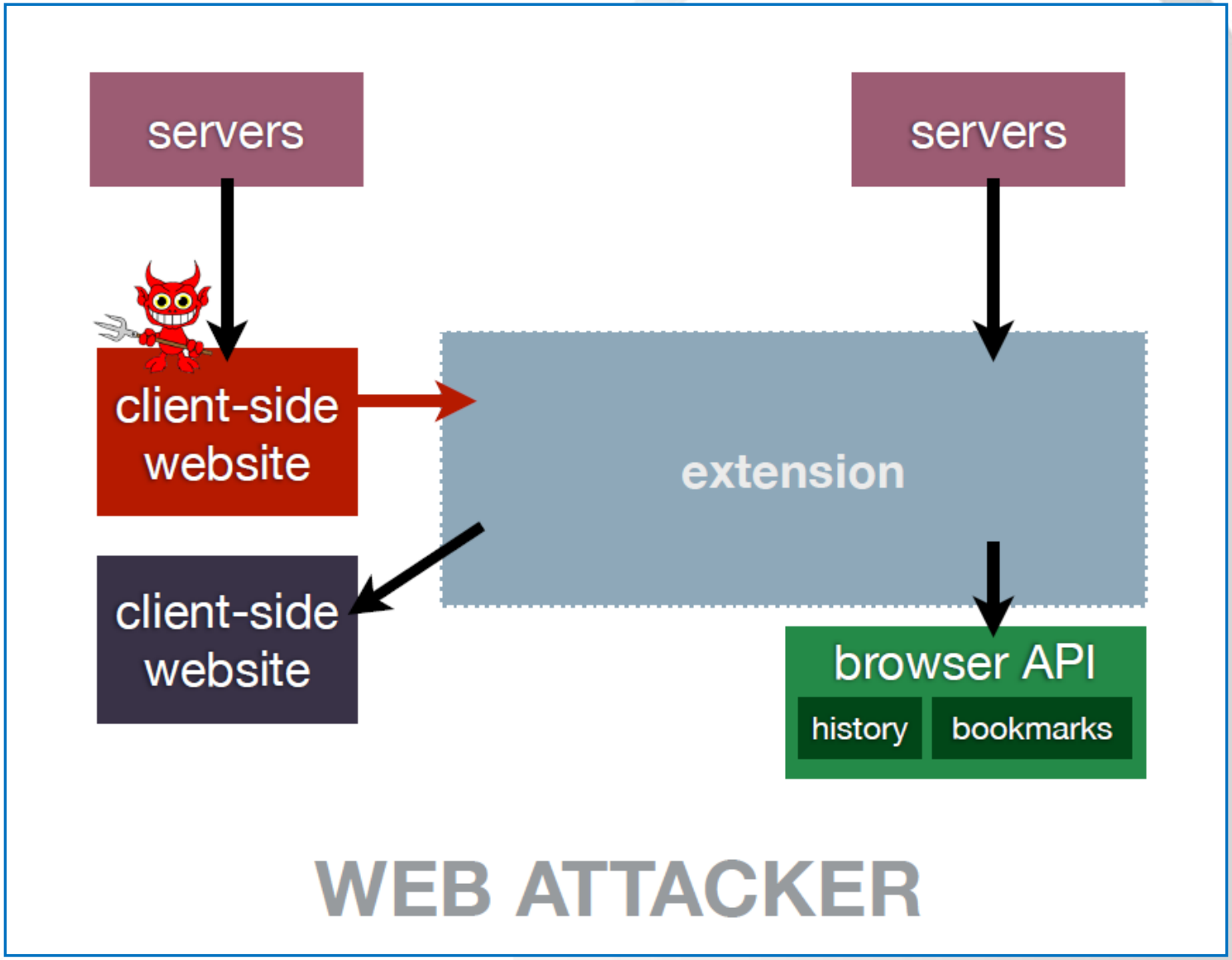
Threat from network attacker



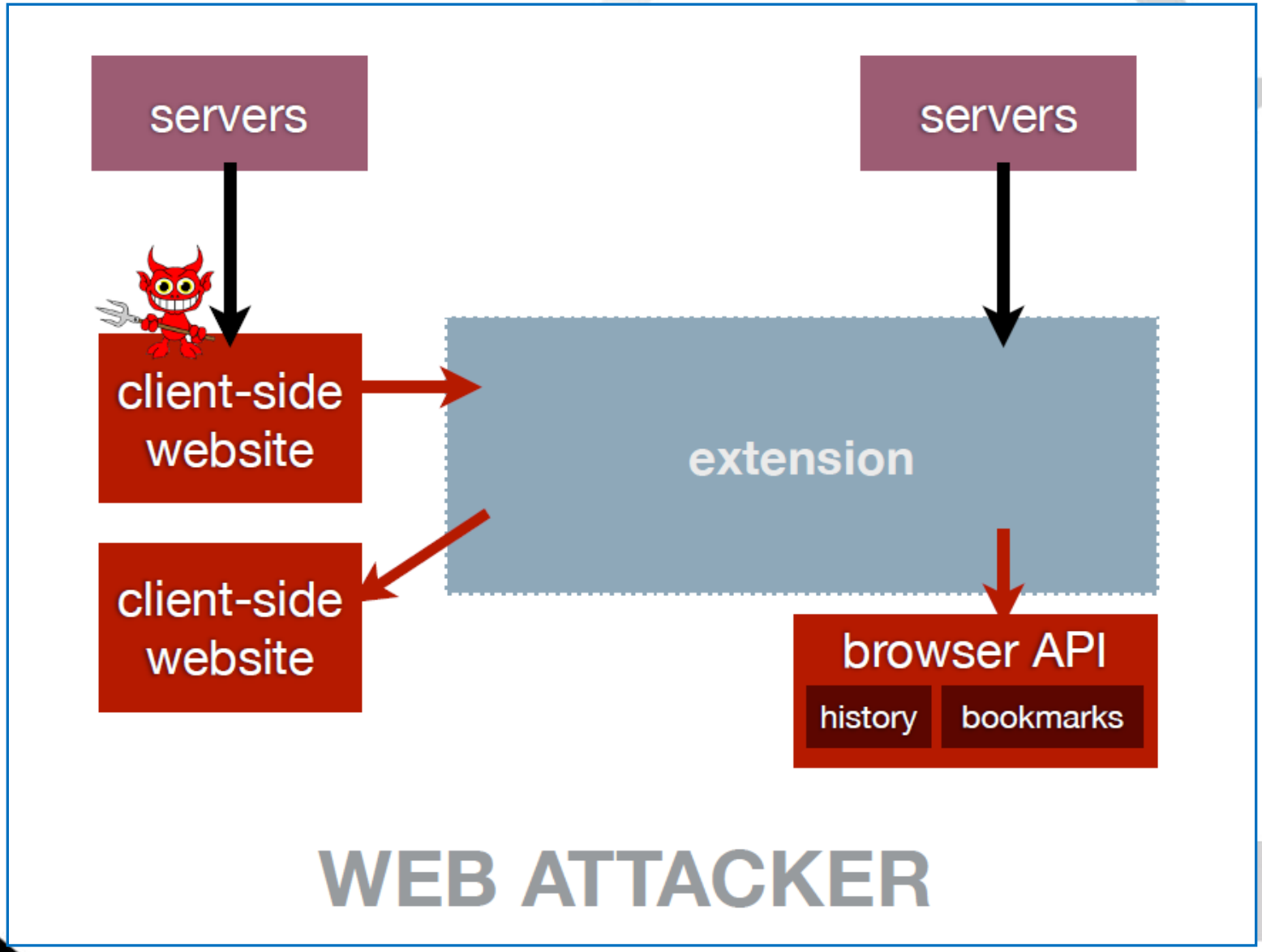
Threat from network attacker



Threat from web attacker



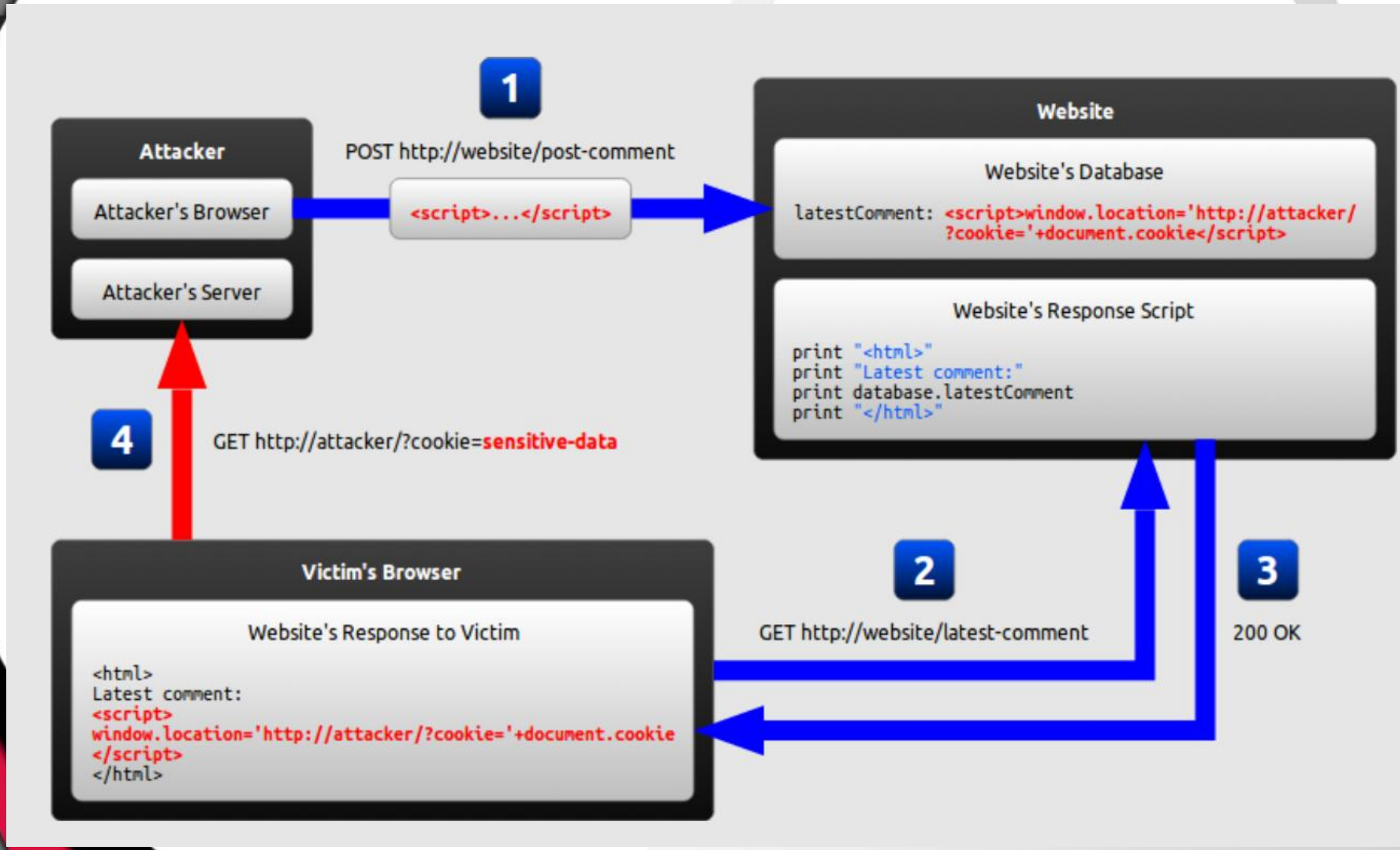
Threat from web attacker



Example_1--XSS Attack

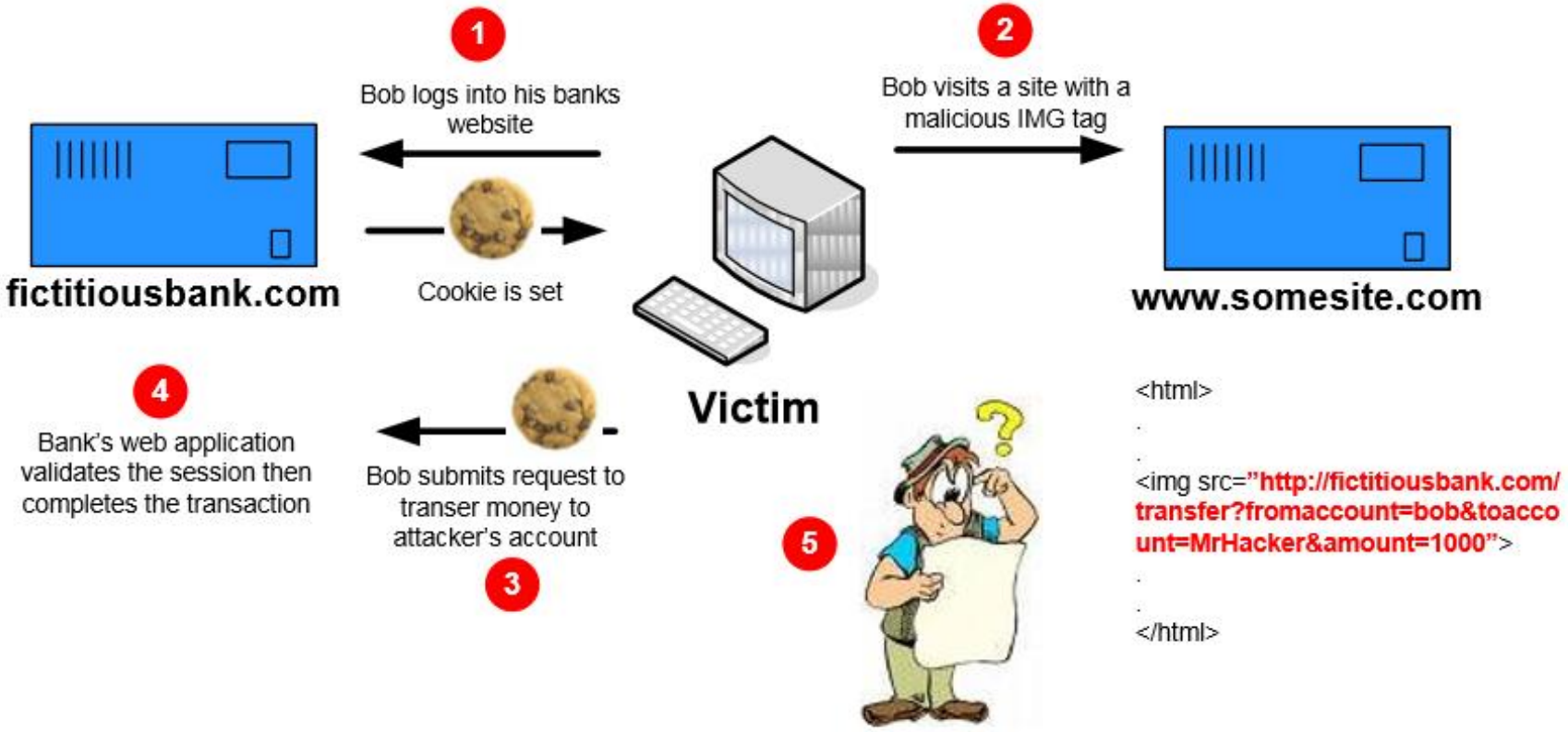
- **Cross-site scripting** : a code injection attack that allows an attacker to execute malicious JavaScript in another user's browser.
- Actors:
  - **The website: (**http://website/)
    - serves HTML pages to users who request them
    - **The website's database** is a database that stores some of the user input included in the website's pages.
  - **The victim:**
    - a normal user of the website who requests pages from it using his browser.
  - **The attacker** :
    - is a malicious user of the website who intends to launch an attack on the victim
    - **The attacker's server: (**http://attacker/) a web server controlled by the attacker
- Goal of the attacker:
  - steal the victim's cookies

Example_1--XSS Attack



Example_2--CSRF Attack

Cross-site Request Forgery: An attack that forces an user's browser to send requests they didn't intend to make

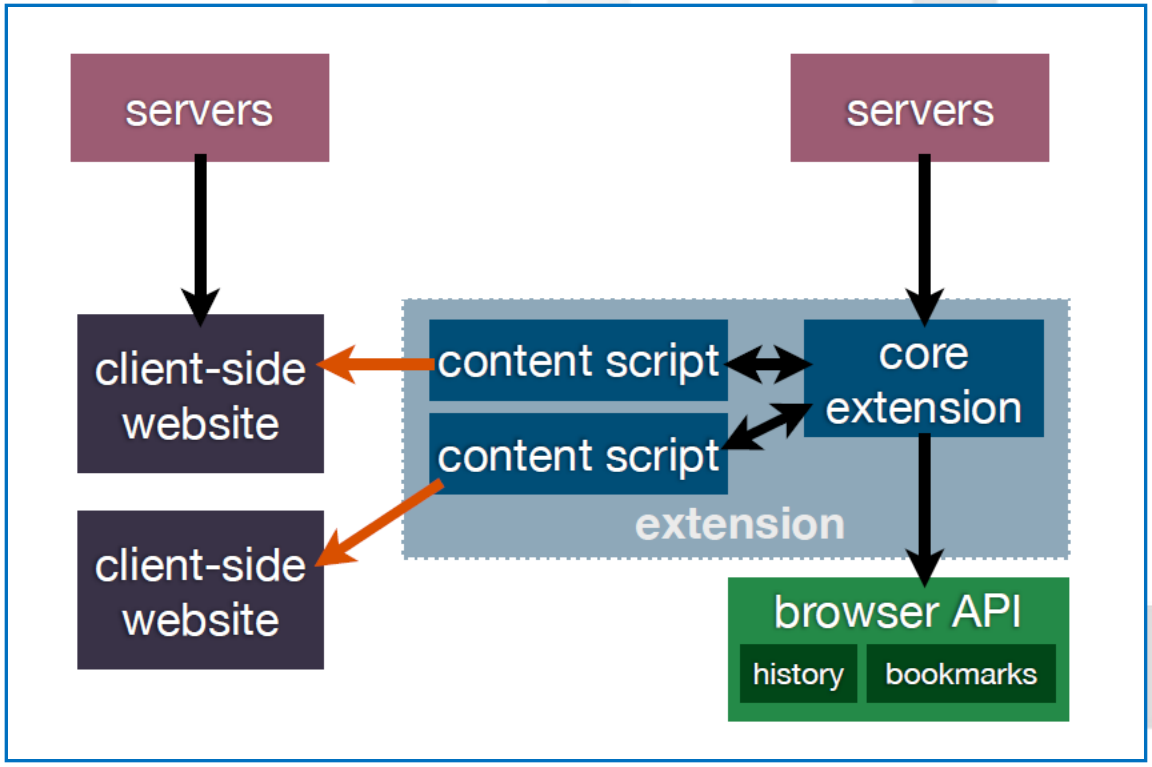


outline

- Chrome extension introduction
- Threats towards extension
- Chrome extension security architecture

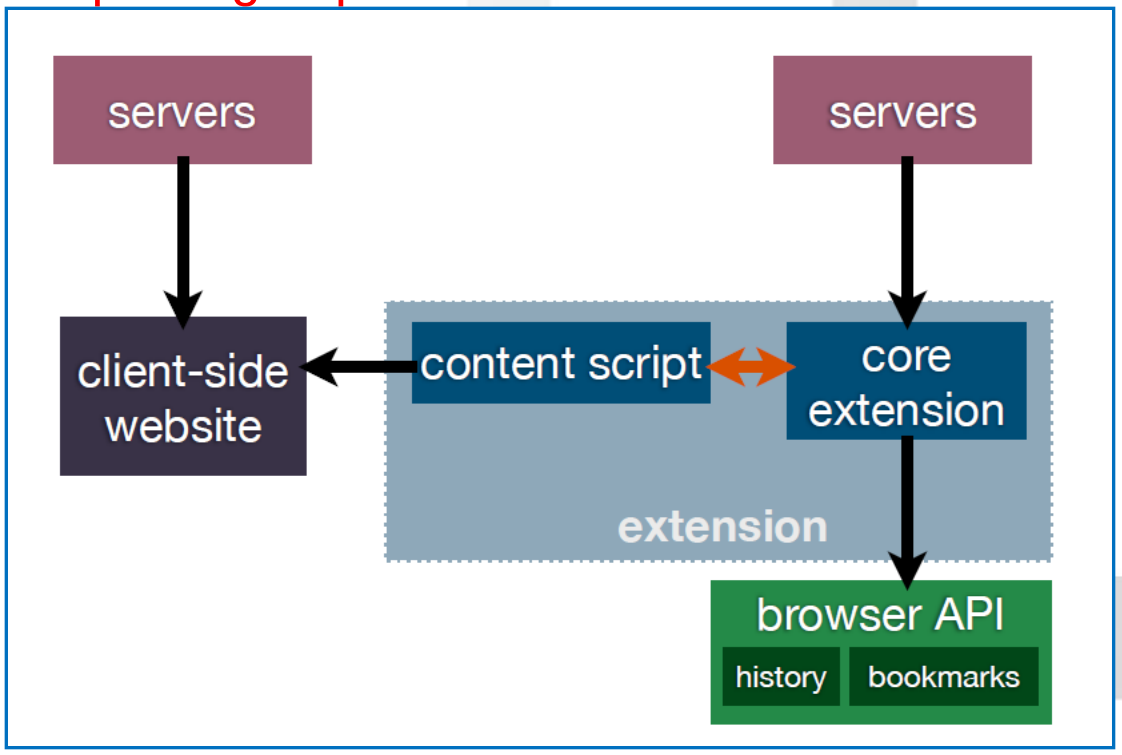
Isolated Worlds

- Content scripts are execute in a special environment called isolated world
  - Have access to the DOM of hosting page
  - Separate javascript heaps
  - No access to variables/functions created by the page
  - Aim to protect content scripts from web attackers



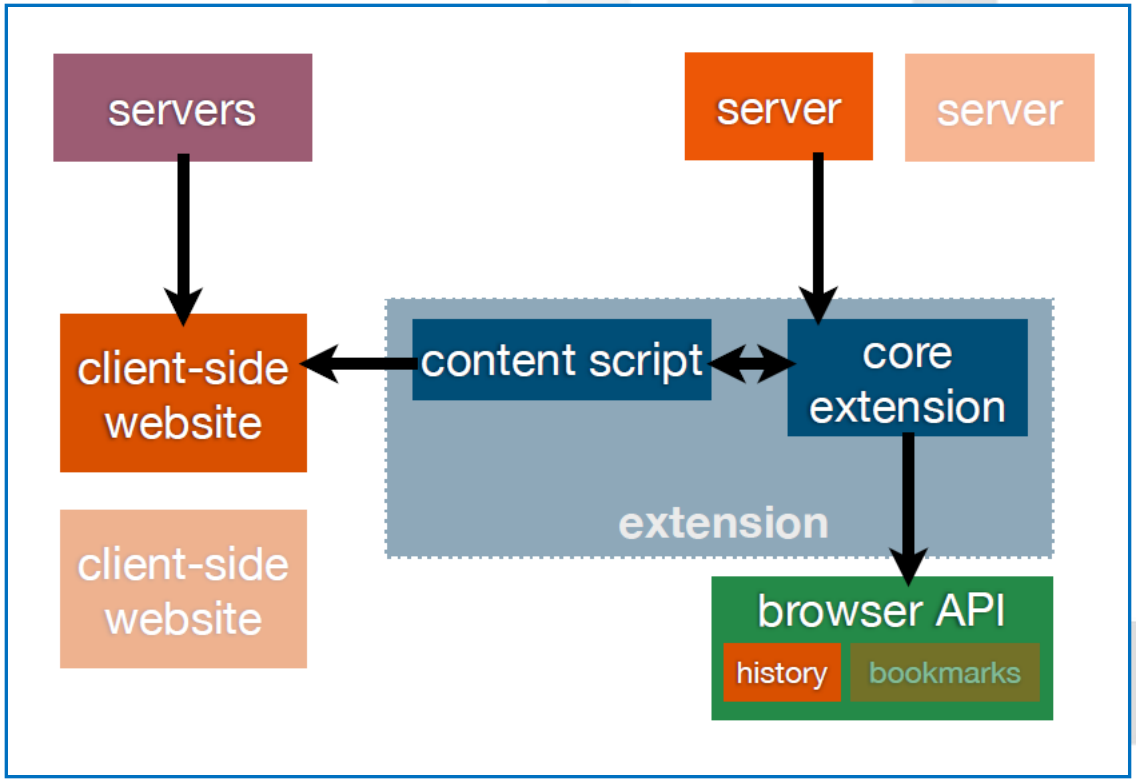
Privilege Separation

- Chrome extension is composed of two types of components:
  - zero or more content scripts & zero or one core extension.
- Content scripts and core extensions run in separate processes, and they communicate by message passing.
- Core extensions can access Chrome's extension API, but content scripts cannot.
- Aim to shield the privileged part of an extension from attackers



Permissions

- By default, extensions cannot use parts of the browser API that impact users' privacy or security.
- A developer must specify the desired permissions in manifest file.
- Content scripts cannot invoke browser APIs
- Aim to mitigate core extension vulnerabilities



Conclusion

- Isolated worlds and Privilege mechanism are highly effective
  - because it prevents common developer errors (i.e., data-as-HTML errors).
- Permissions can have a significant positive impact on system security
  - developers of vulnerable extensions can use permissions well enough to reduce the scope of their vulnerabilities

Thanks !

Q & A