

# HTML Injection Attacks

# HTML Injection

- When hosting static pages
  - You control all the content
  - Limited opportunity for attackers
- When hosting user-submitted content
  - You lose that control
  - Must protect against attacks
  - **Never trust your users!!**

**Never Trust Your Users!**

**Never Trust Your Users!**

**Seriously. NEVER.**

# Never Trust Your Users

- You may want to think your users will all act in good faith
- For most users, this may be true

# Never Trust Your Users

- You may want to think your users will all act in good faith
  - For most users, this may be true
- Besides your intended users, who else can access your app?

# Never Trust Your Users

- You may want to think your users will all act in good faith
  - For most users, this may be true
- Besides your intended users, who else can access your app?
  - **EVERYONE!**

# Never Trust Your Users

- Do you trust **literally everyone**??



# HTML Injection

- You are now handling user data and sending it to other users (Through chat messages)
- You're building a form that accepts user data and serves it to all other users
- What happens when a user enters this in chat:
  - "`<script>maliciousFunction()</script>`"

# HTML Injection

- "`<script>maliciousFunction()</script>`"
- This attack is called an HTML injection attack
  - This string is uploaded to your server
  - Your server stores this string
  - Your server sends this string to all users who use your app
  - Their browsers render the injected HTML
  - Their browsers run the injected JS

# HTML Injection

- Lucky for us, Preventing this attack is very simple

# HTML Injection

- To prevent this attack:
  - Escape HTML when handling user submitted data
- Escape HTML
  - Replace `&`, `<`, and `>` with their HTML escaped characters
    - `'&'` -> `&amp;`;
    - `'<'` -> `&lt;`;
    - `'>'` -> `&gt;`;

# HTML Injection

- The escaped characters `&amp;`, `&lt;`, `&gt;` will be rendered as characters by the browser
- Browser does not treat these as HTML

# HTML Injection

- Replace &, <, and > with their HTML escaped characters
- `<script>maliciousFunction()</script>`
  - becomes
- `&lt;script&gt;maliciousFunction()&lt;/script&gt;`
  - and is rendered as a string instead of interpreted as HTML

# HTML Injection

- Replace &, <, and > with their HTML escaped characters
- Order is important!
  - Always escape & first
  - If & is escaped last you'll get:
- `&amp;lt;script&amp;gt;maliciousFunction()&amp;lt;/script&amp;gt;`
- Which will not render the way you intended

Demos