ATTACKING THE FRONTEND

CodingWithCallum™ Lightning Talk

COMMON ISSUES

- Code injection
  - How does the web app handle unexpected data
- XSS
  - "Cross Site Scripting"
  - Attacker submitted code run in browser
  - CSP by itself cannot prevent XSS
- HTML elements can run code!

REACT

- Context sensitive output encoding
    - out of the box
    - until you use something with the word `dangerous` in front of it
- React will protect you from yourself

DANGEROUSLYSETINNERHTML

- Don't use this + the linter will yell at you if you do
- Needs DOMPurify if you need to use it
  - `{dangerouslySetInnerHTML: DOMPurify.sanitize({html})}`

ESCAPE HATCHES

- Bypass react and access native DOM APIs
- Direct DOM Manipulation
- Good news: React is deprecating this
    - should be disallowed
        - `findDOMNode`
        - `innerHTML`
    - `createRef` is not

ENCODING URLS

- Avoid taking full URL as an input
- Do URL sanitisation
    - Nextjs does this for us
- Should allowlist certain urls

RESOURCE URLS

- Javascript & Resource URLs can be a potential sink
    - are being disallowed from React 17+
    - `data.` still runs

BEST PRACTICES

- "If you are going to the DOM directly, talk to security"
- CSRF
- Cookies should have `samesite` set (or better be secure)

RESOURCES

- ReactVulna is a deliberately vulnerable app you can play around with
- React has a guide on escape hatches
- Its-fine A collection of escape hatches exploring
- `React.__SECRET_INTERNALS_DO_NOT`
- `_USE_OR_YOU_WILL_BE_FIRED`