



Building and Breaking: Web Applications

Holly Grace Williams
Managing Director Akimbo Core





Hi, my name is @HollyGraceful

~~I break into computers for a living.~~

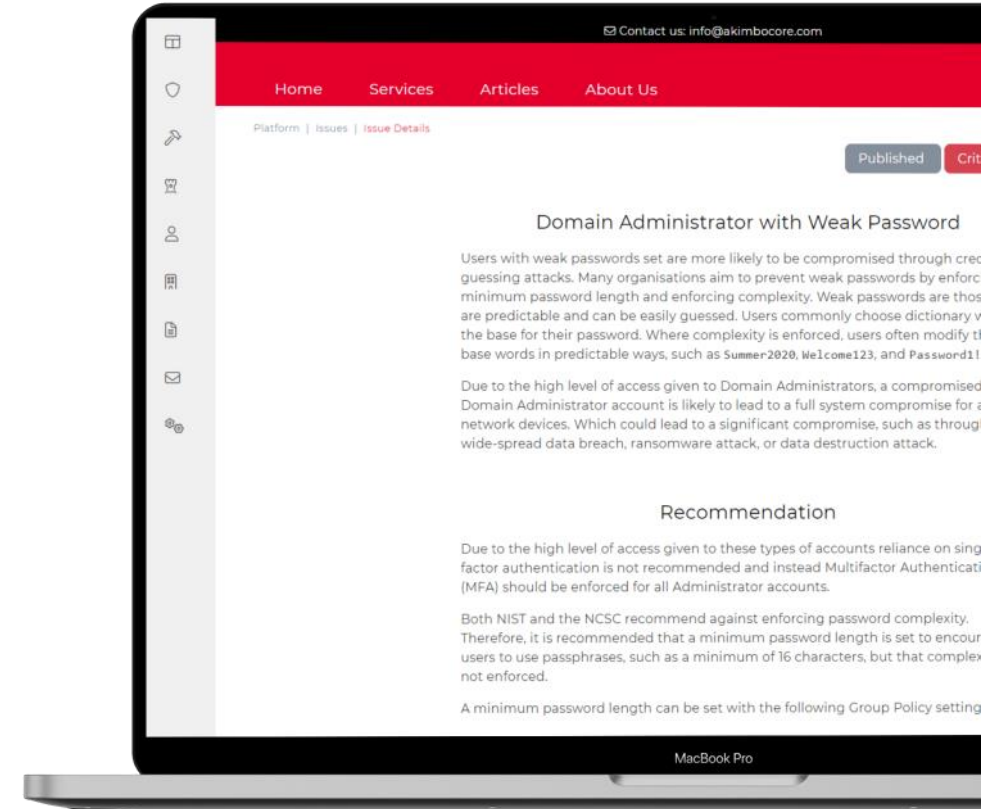
I write code that breaks into computers for a living.



Demonstrating Real-world Risk



"Penetration Testing is a **human-led, scope-restricted, time-limited** security assessment aiming to exploit vulnerabilities to demonstrate their real-world risk."



Demonstrating Real-world Risk

AKIMBOCORE

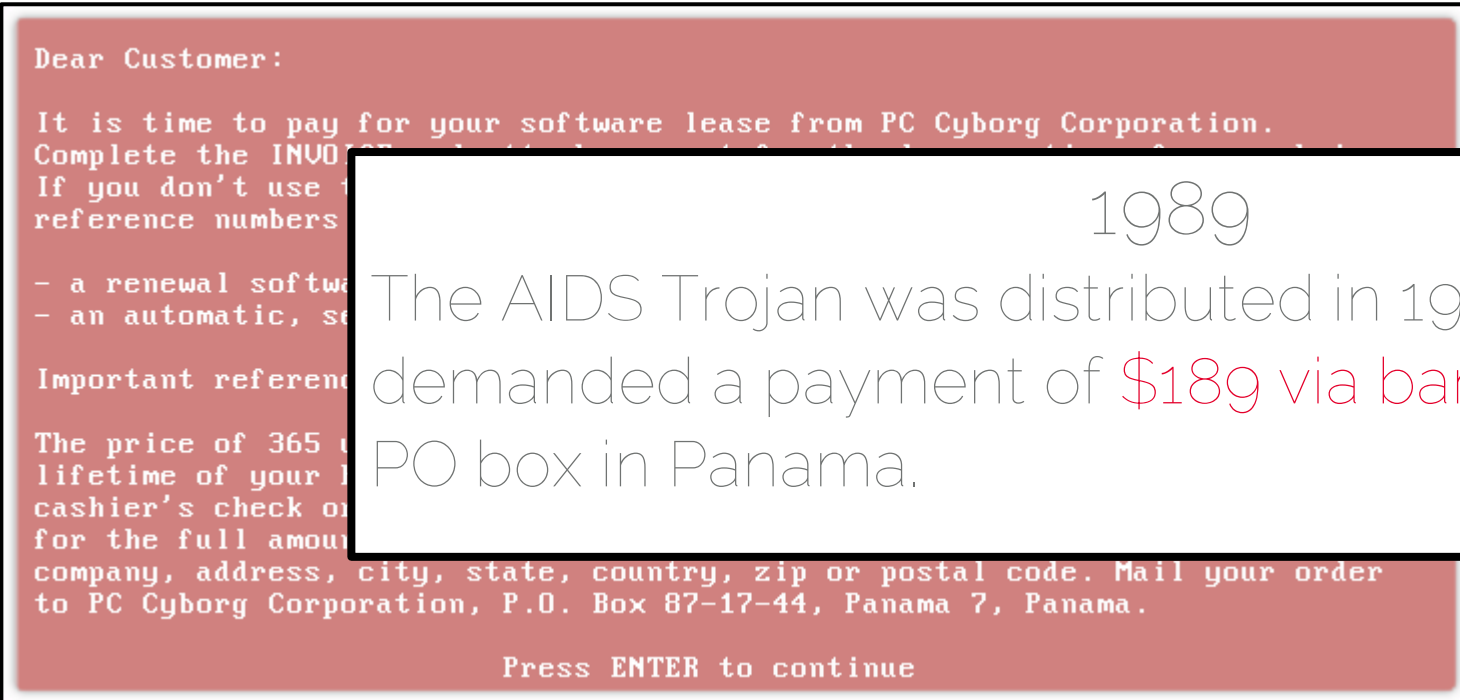


Penetration Testing often focuses too heavily on "cool" or "interesting" vulnerabilities instead of what's best for improving the overall security stance of the target organisation.

Let's talk about breaking the attack chain...

"Ransomware Attack"

Ransomware: AIDS Trojan



1989

The AIDS Trojan was distributed in 1989 and demanded a payment of \$189 via banker's draft to a PO box in Panama.

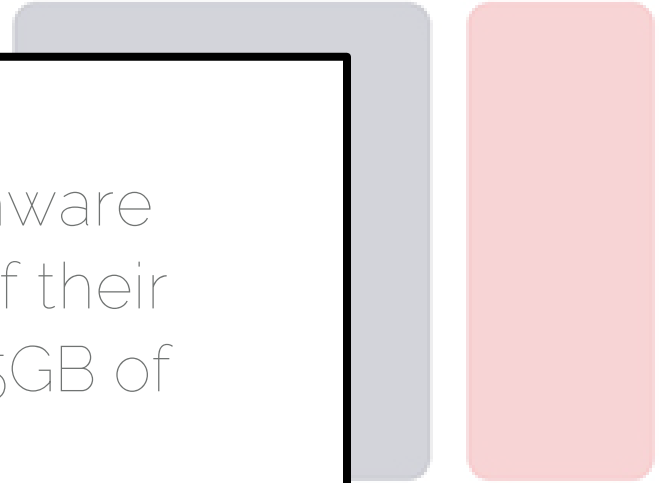


"Ransomware Attack"

Ransomware: REvil aka Sodinokibi

```
--=== Welcome. Again. ===--  
  
[+] Whats Happen? [+]  
  
Your files are encrypted  
your computer has ex  
By the way, everyth  
instructions. Other  
  
[+] What guarantees  
  
Its just a business  
getting benefits. I  
cooperate  
with us. Its not in  
To check the abilit  
decrypt one file fo  
If you will not cooperate with our service - for us, its does not matter. But you  
will lose  
your time and data, cause just we have the private key. In practise - time is much  
more valuable than money.
```

2020
Travelex paid \$2.3 million dollars to ransomware
attackers who encrypted significant parts of their
network, deleted backups, and exfiltrated 5GB of
personal data.



Travelex

Demonstrating Real-world Risk

AKIMBOCORE



Best Practice often conflicts with other Best Practice, which can lead companies to make poor choices or feel unsure as to how to harden their systems.

"Best Practice"

Passwords should be complex

PCI DSS 8.2.3 Passwords must contain both numeric and alphabetic characters¹.

Change your passwords regularly

PCI 8.2.4 Change user passwords/passphrases at least once every 90 days

PCI DSS 3.2.1, https://www.pcisecuritystandards.org/document_library

"Best Practice"

~~Passwords should be complex~~

“Do not use complexity requirements. [...] the NCSC do **not** recommend the use of complexity requirements when implementing user generated passwords”

~~Change your passwords regularly~~

“Don't enforce regular password expiry. Regular password changing harms rather than improves security” “Regular password expiry is a common requirement in many security policies. However, in the Password Guidance published **in 2015, we explicitly advised against it.**”

<https://www.ncsc.gov.uk/blog-post/problems-forcing-regular-password-expiry>

<https://www.ncsc.gov.uk/collection/passwords/updating-your-approach>

<https://pages.nist.gov/800-63-3/sp800-63b.html>



⚠ Invalid password. Password must contain at least 1 symbol, 1 number, 1 upper-case, 1 lower-case and be 8 characters long.

Invalid password.

Password **must** contain at least 1 symbol, 1 number, 1 upper-case, 1 lower-case and **be 8 characters long**.





Enter your new password:(6-15 characters including both letters and numbers)



 

* We're sorry, but the details you entered aren't in the right format.
Please check them and try again.

Re-enter your new password:

* Please provide this information.

 Cancel 

Enter your new password:(**6-15 characters including both letters and numbers**)

OWASP Top 10

- A1 Injection
- A2 Broken Authentication
- A3 Sensitive Data Exposure
- A4 XML External Entities
- A5 Broken Access Control
- A6 Security Misconfiguration
- A7 Cross-site Scripting**
- A8 Insecure Deserialization
- A9 Using Components with Known Vulnerabilities
- A10 Insufficient Logging and Monitoring

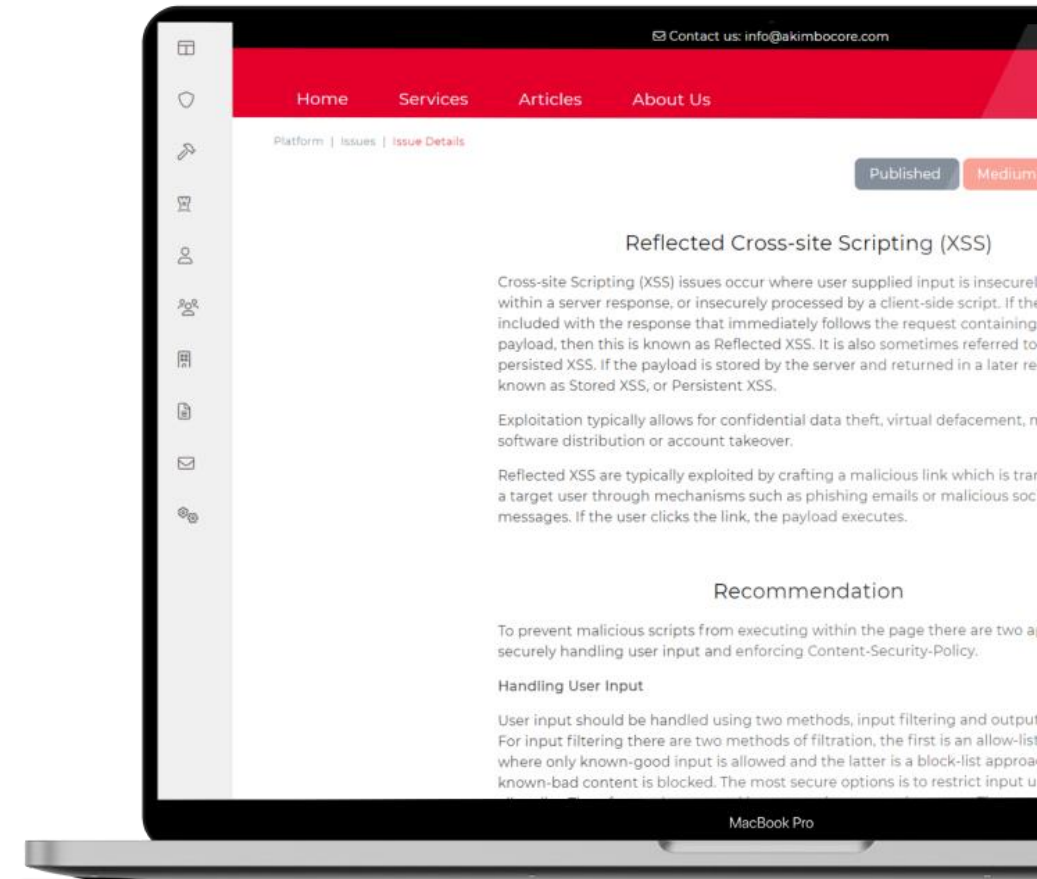
Vulnerability Example

Cross-site Scripting

How common is XSS?

XSS was the most common vulnerability in 2019, accounting for 18% of discovered vulnerabilities

— State of Open Source Security, Snyk

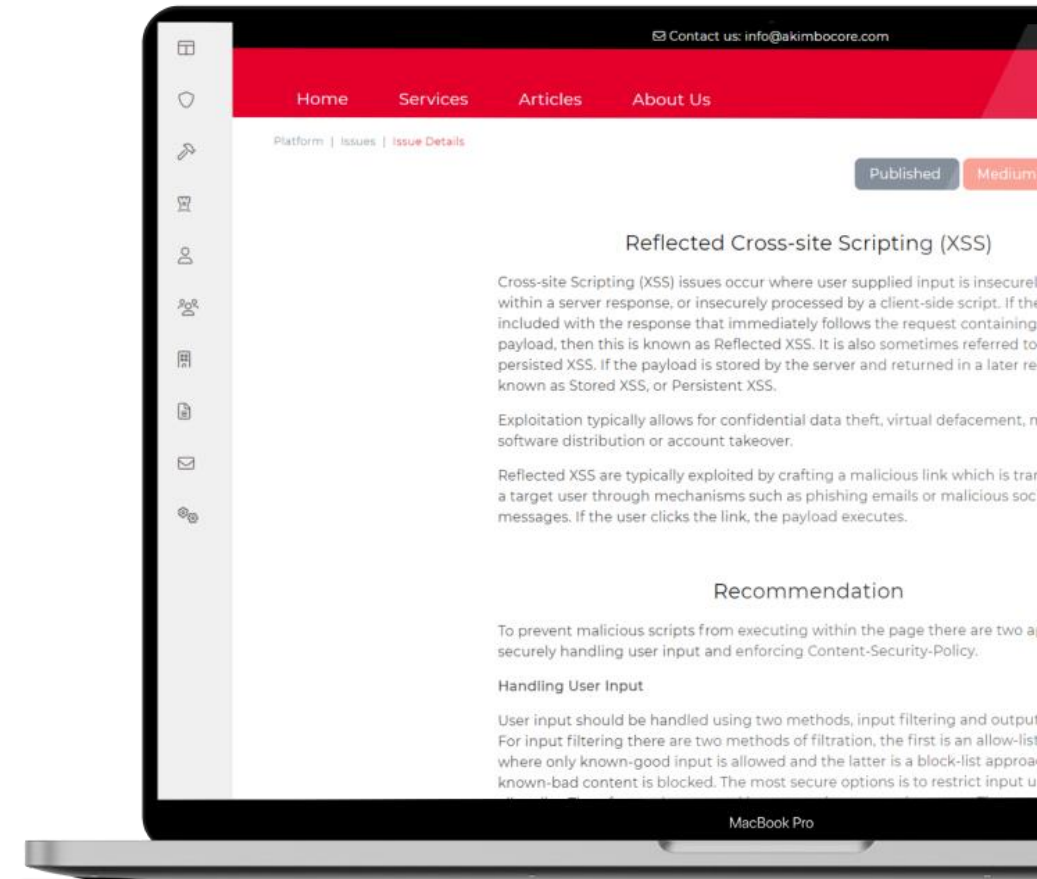


Vulnerability Example

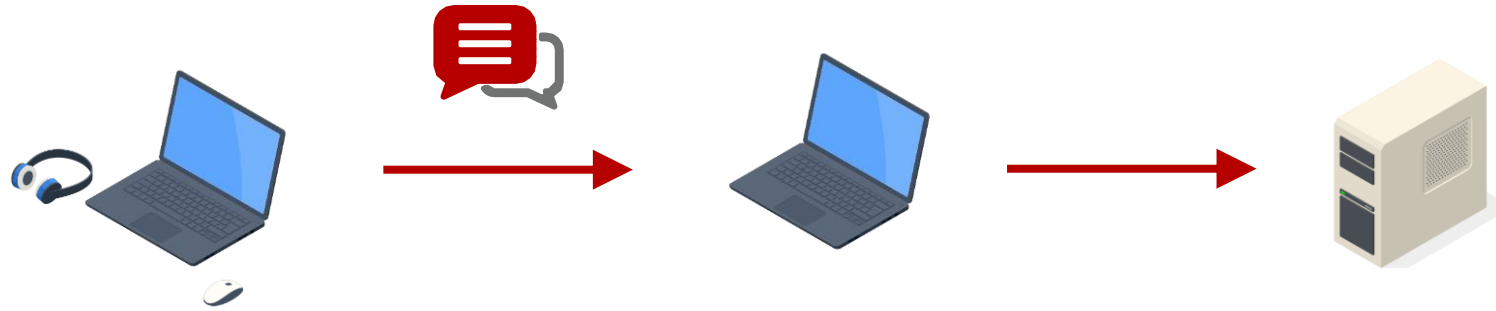
AKIMBOCORE



"Cross-site Scripting (XSS) issues occur where user supplied input is **insecurely included within a server response**, or insecurely processed by a client-side script."



Vulnerability Example



Reflected Cross-site Scripting (XSS)

1. The threat actor sends a payload containing dynamic scripts to a vulnerable web application.
2. At some point in the future, the user accesses the page containing the payload and script execution occurs within their browser.

How common is XSS?

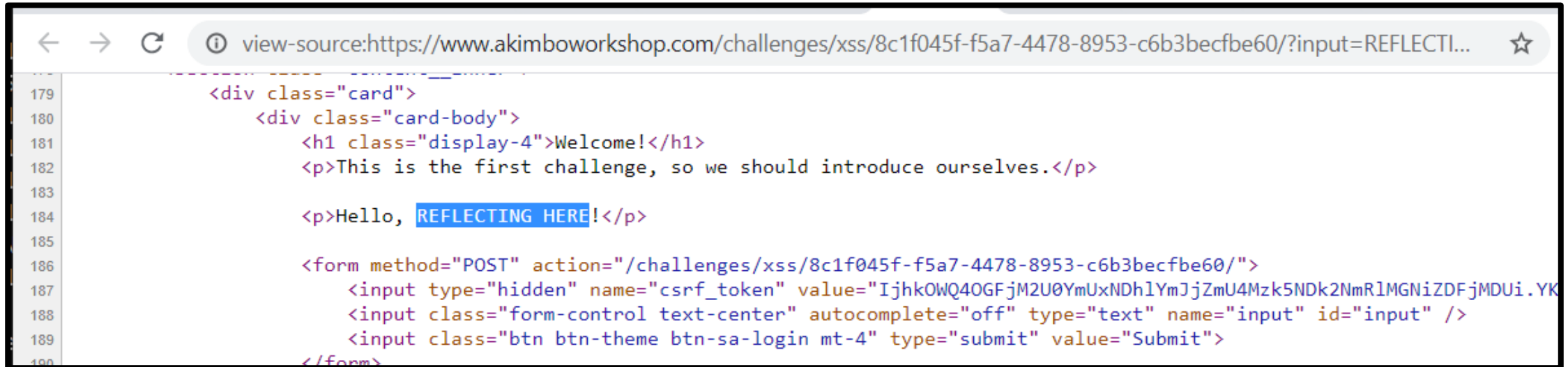
XSS was the most common vulnerability in 2019, accounting for 18% of discovered vulnerabilities

– State of Open Source Security, Snyk

Vulnerability Example

Discovery

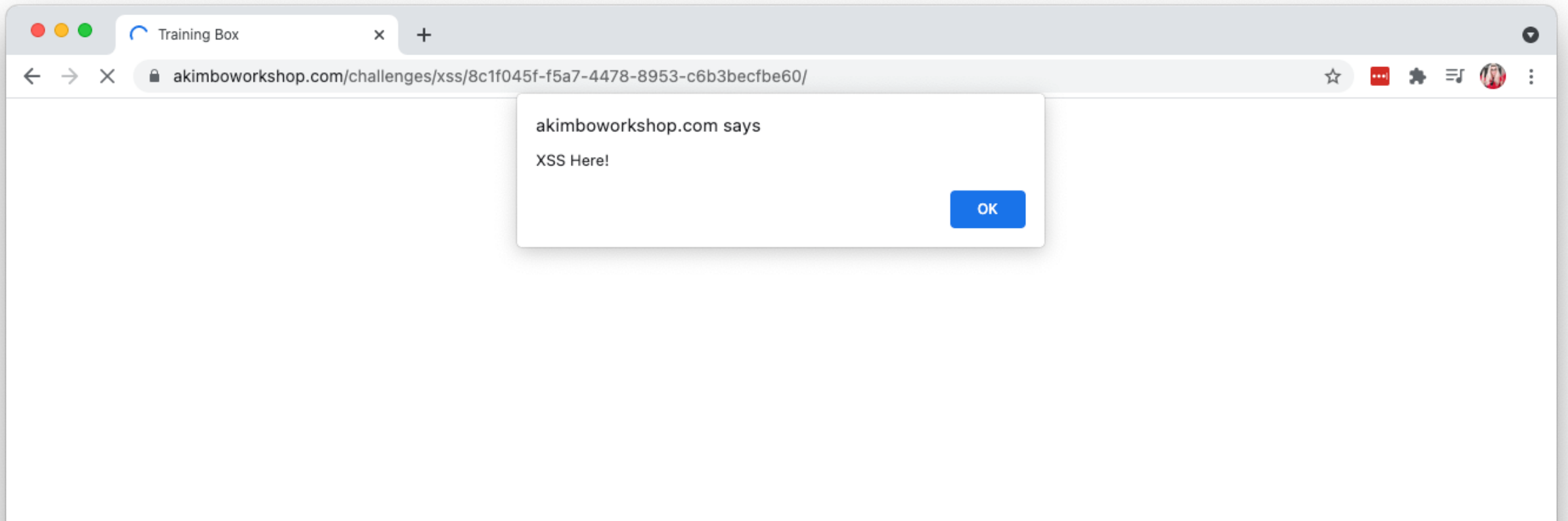
The payload required to execute JavaScript will differ depending on where in the page you're being reflected.



```
179     <div class="card">
180         <div class="card-body">
181             <h1 class="display-4">Welcome!</h1>
182             <p>This is the first challenge, so we should introduce ourselves.</p>
183
184             <p>Hello, REFLECTING HERE!</p>
185
186             <form method="POST" action="/challenges/xss/8c1f045f-f5a7-4478-8953-c6b3becfbe60/">
187                 <input type="hidden" name="csrf_token" value="IjhkOWQ40GFjM2U0YmUxNDhlYmJjZmU4Mzk5NDk2NmRlMGNiZDFjMDUi.YK
188                 <input class="form-control text-center" autocomplete="off" type="text" name="input" id="input" />
189                 <input class="btn btn-theme btn-sa-login mt-4" type="submit" value="Submit">
190             </form>
```

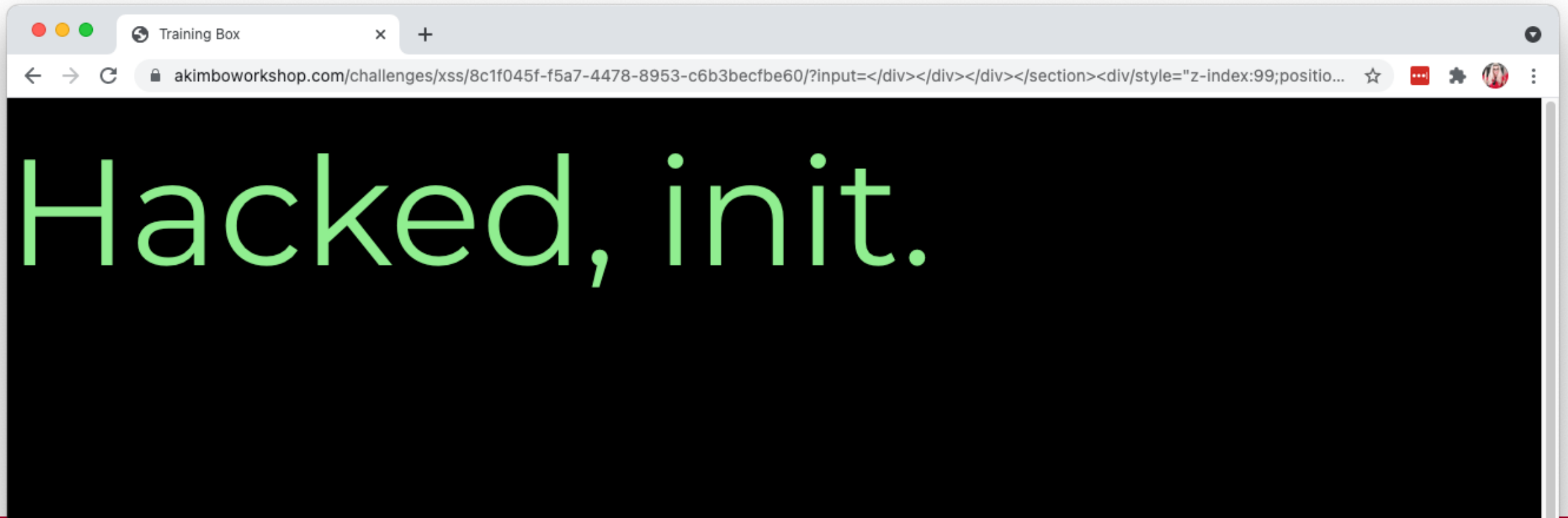

Hardening Example

```
<script>alert('XSS Here!')</script>
```

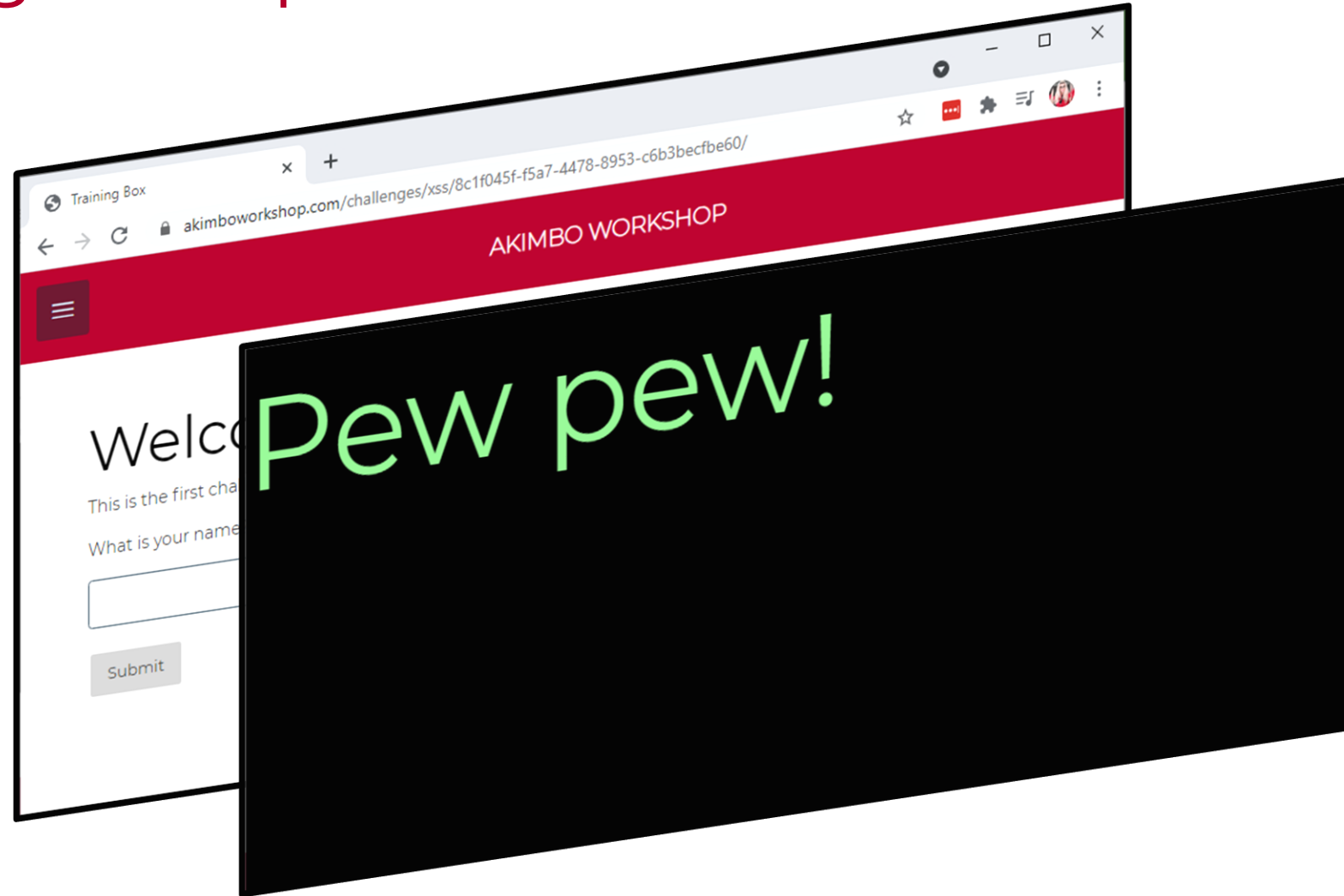


Hardening Example

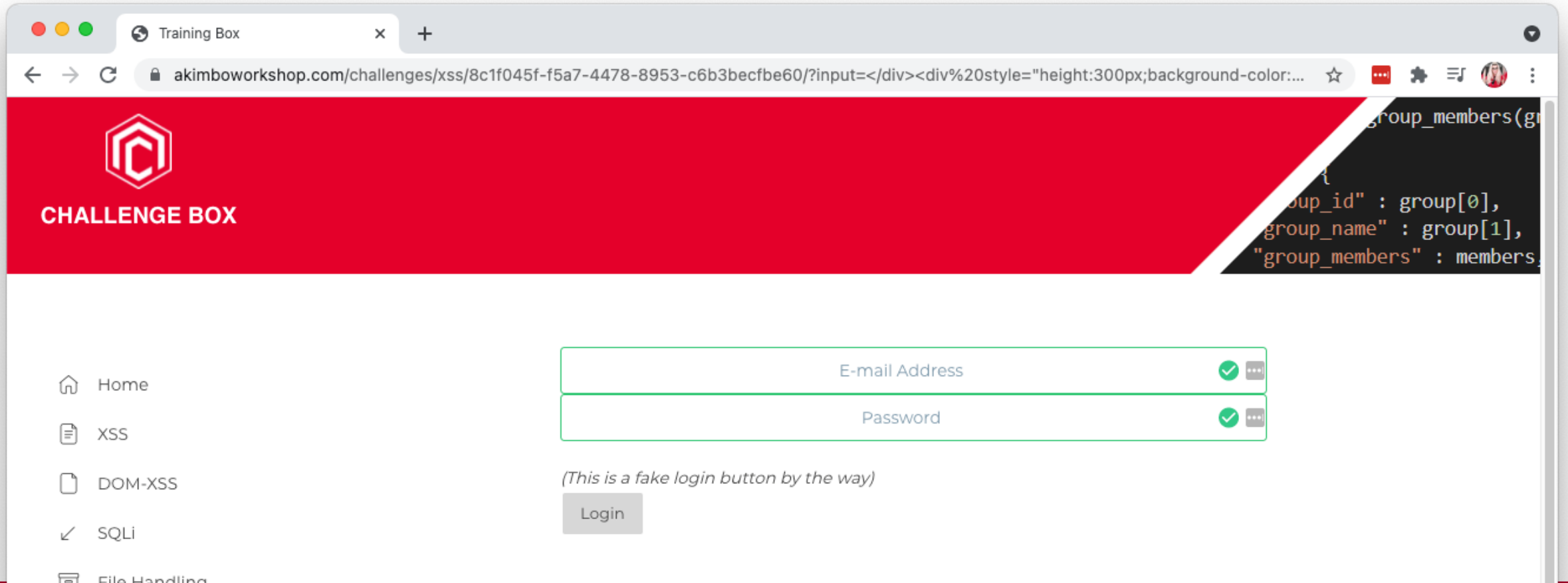
```
</div></div></div></section><div/style="z-index:2;height:1080px;
width:1920px;position:absolute; top:0;left:0;background-color:black;
color:lightgreen;font-size:96pt">Hacked, Init</div>
```



Hardening Example



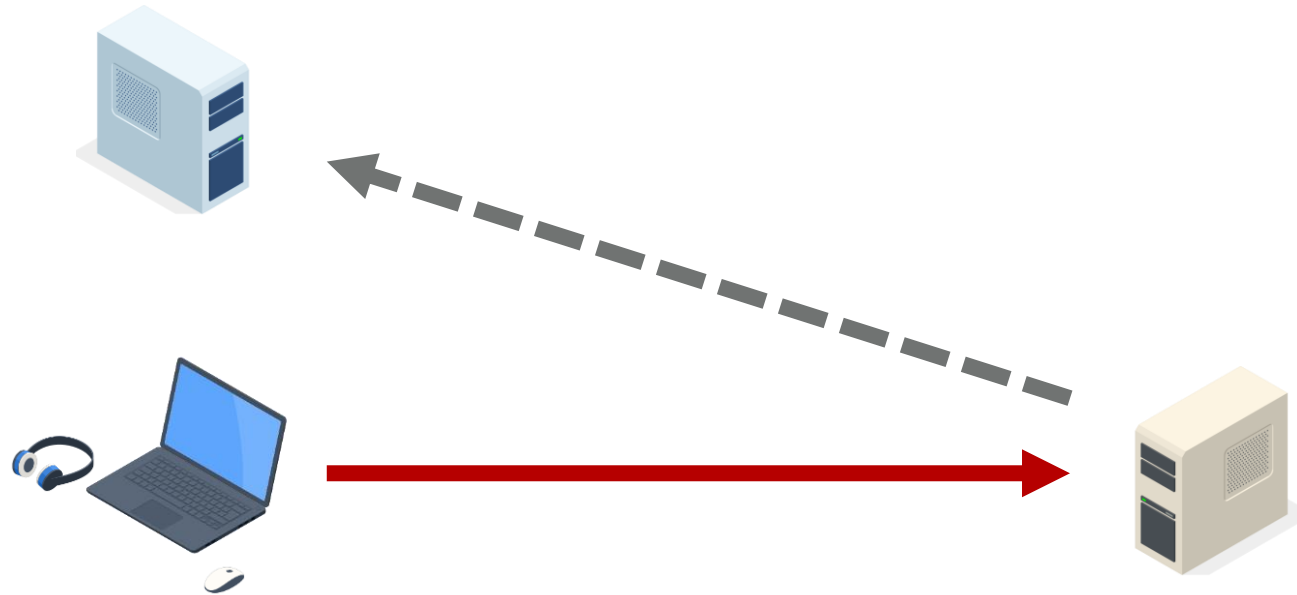
Hardening Example



The screenshot shows a web browser window with the title "Training Box" and the URL `akimboworkshop.com/challenges/xss/8c1f045f-f5a7-4478-8953-c6b3becfbe60/?input=</div><div%20style="height:300px;background-color:...`. The page features a red header with the AkimboCore logo and the text "CHALLENGE BOX". On the left, there is a navigation menu with items: Home, XSS, DOM-XSS, SQLi, and File Handling. The main content area contains a login form with two input fields: "E-mail Address" and "Password", both marked with green checkmarks. Below the form is a grey "Login" button and a note: "(This is a fake login button by the way)". On the right side, a terminal window displays a JSON object: `{ "group_id" : group[0], "group_name" : group[1], "group_members" : members,`

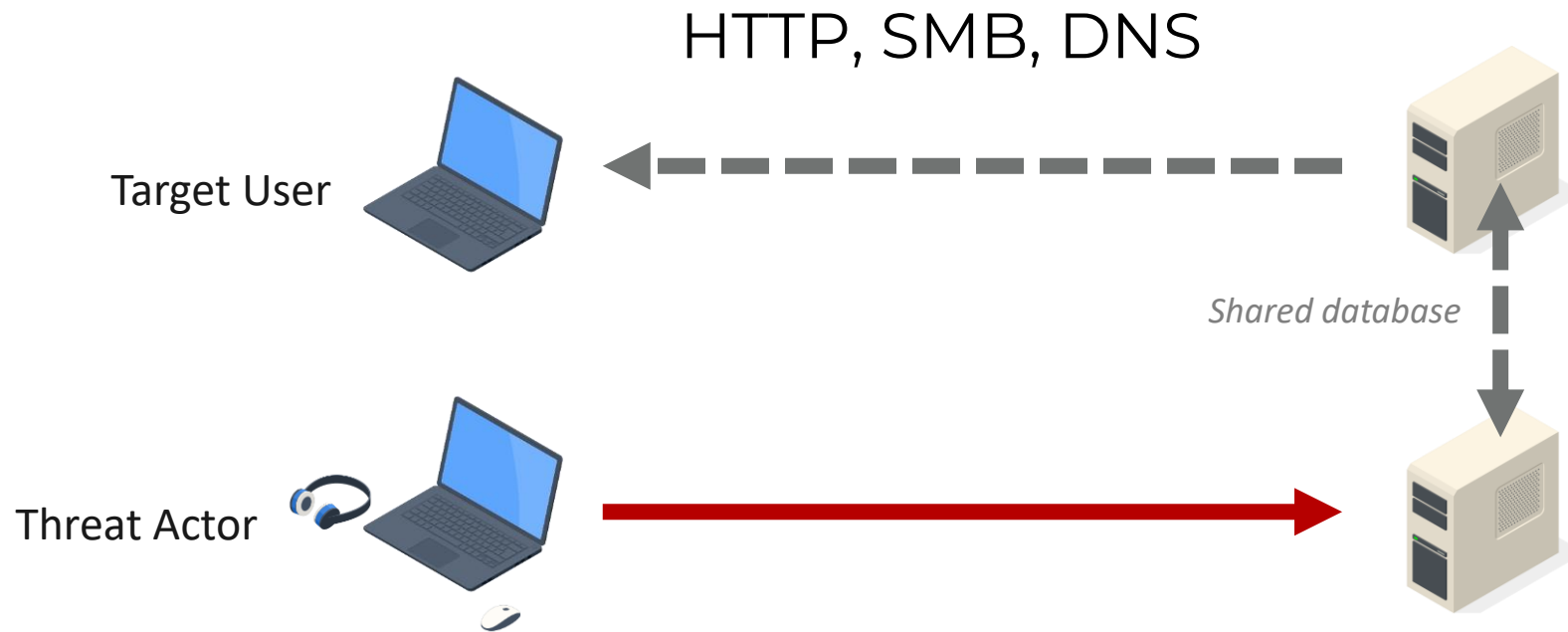
Out-of-Band Exploitation

Out-of-band Exploitation refers to exploiting an application and extracting information through a mechanism other than directly through front-end responses.



Out-of-Band Exploitation

It's also possible to use this methodology to infer the presence of additional target applications and target them. Exfiltration can occur using a range of protocols:



Vulnerability Example

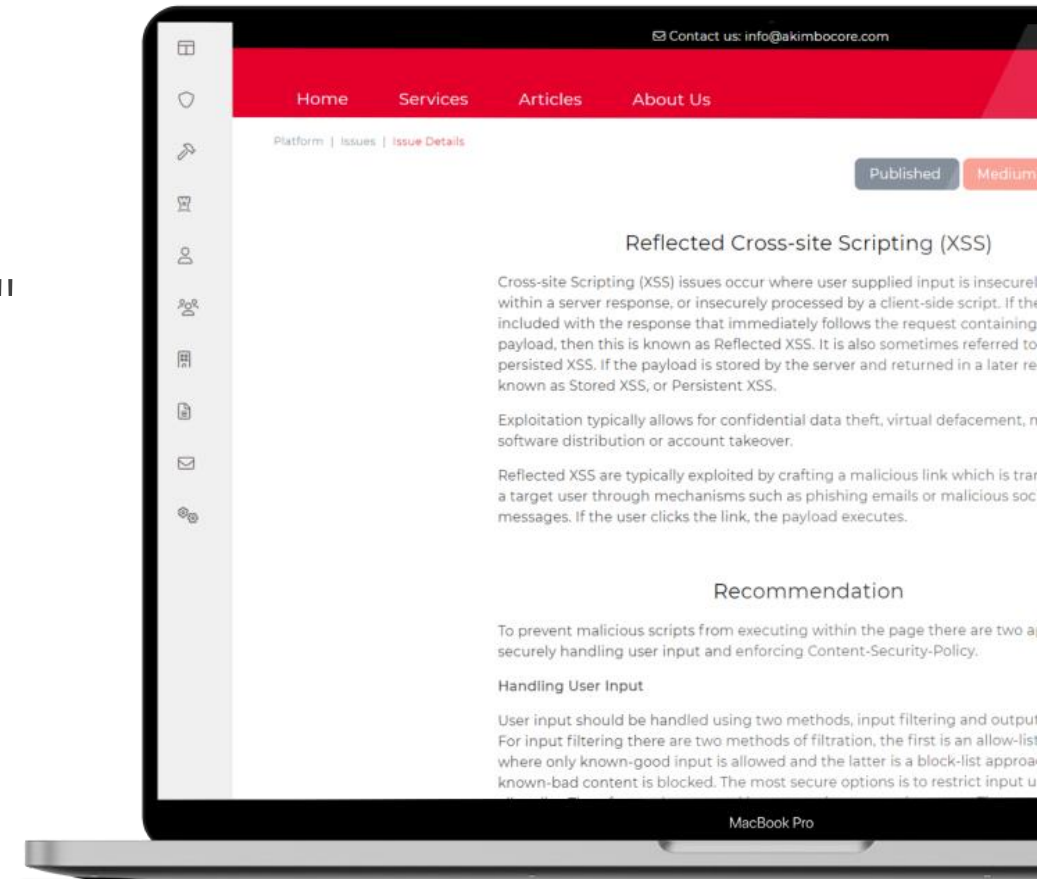
AKIMBOCORE



"Cross-site Scripting (XSS) issues occur where user supplied input is **insecurely included within a server response**, or insecurely processed by a client-side script."

```
<script>alert('XSS Here!')</script>
```

```
&lt;script&gt;alert(&apos;XSS Here&apos;)&lt;script&gt;
```



Hardening Example

CSP is a protection built into web browsers that allows a developer to **restrict the source of scripts** and resources to only approved locations.

It is enabled through a server response header.

<https://akimboCore.com/article/content-security-policy/>

Hardening Example

```
Content-Security-Policy: default-src 'self';  
script-src 'self' https://js.example.com;  
style-src 'self' https://css.example.com;
```

<https://akimbo.com/article/content-security-policy/>

Hardening Example

AKIMBOCORE



```
connect-src 'self' blob: https://*.giphy.com https://*.pscp.tv https://*.video.pscp.tv https://*.twimg.com https://api.twitter.com https://api-stream.twitter.com https://ads-api.twitter.com https://aa.twitter.com https://caps.twitter.com https://media.riffsy.com https://pay.twitter.com https://sentry.io https://ton.twitter.com https://twitter.com https://upload.twitter.com https://www.google-analytics.com https://app.link https://api2.branch.io https://bnc.lt wss://*.pscp.tv https://vmap.snappytv.com https://vmapstage.snappytv.com https://vmaprel.snappytv.com https://vmap.grabyo.com https://dhdsnappytv-vh.akamaihd.net https://pdhdsnappytv-vh.akamaihd.net https://mdhdsnappytv-vh.akamaihd.net https://mdhdsnappytv-vh.akamaihd.net https://mpdhdsnappytv-vh.akamaihd.net https://mmdhdsnappytv-vh.akamaihd.net https://mdhdsnappytv-vh.akamaihd.net https://mpdhdsnappytv-vh.akamaihd.net https://mmdhdsnappytv-vh.akamaihd.net https://dwo3ckksxlb0v.cloudfront.net ; default-src 'self'; form-action 'self' https://twitter.com https://*.twitter.com; font-src 'self' https://*.twimg.com; frame-src 'self' https://twitter.com https://mobile.twitter.com https://pay.twitter.com https://cards-frame.twitter.com https://accounts.google.com/; img-src 'self' blob: data: https://*.cdn.twitter.com https://ton.twitter.com https://*.twimg.com https://analytics.twitter.com https://cm.g.doubleclick.net https://www.google-analytics.com https://www.periscope.tv https://www.pscp.tv https://media.riffsy.com https://*.giphy.com https://*.pscp.tv https://*.periscope.tv https://prod-periscope-profile.s3-us-west-2.amazonaws.com https://platform-lookaside.fbsbx.com https://scontent.xx.fbcdn.net https://scontent-sea1-1.xx.fbcdn.net https://*.googleusercontent.com https://imgix.revue.co; manifest-src 'self'; media-src 'self' blob: https://twitter.com https://*.twimg.com https://*.vine.co https://*.pscp.tv https://*.video.pscp.tv https://*.giphy.com https://media.riffsy.com https://dhdsnappytv-vh.akamaihd.net https://pdhdsnappytv-vh.akamaihd.net https://mdhdsnappytv-vh.akamaihd.net https://mdhdsnappytv-vh.akamaihd.net https://mpdhdsnappytv-vh.akamaihd.net https://mmdhdsnappytv-vh.akamaihd.net https://mdhdsnappytv-vh.akamaihd.net https://mpdhdsnappytv-vh.akamaihd.net https://mmdhdsnappytv-vh.akamaihd.net https://dwo3ckksxlb0v.cloudfront.net; object-src 'none'; script-src 'self' 'unsafe-inline' https://*.twimg.com https://www.google-analytics.com https://twitter.com https://app.link https://accounts.google.com/gsi/client https://appleid.cdn-apple.com/appleauth/static/jsapi/appleid/1/en_US/appleid.auth.js 'nonce-NWQ5ZDljZTgtOTM5NC00NDE1LTlmMTgtZjBiOGQ1NTAyNTIz'; style-src 'self' 'unsafe-inline' https://accounts.google.com/gsi/style https://*.twimg.com; worker-src 'self' blob:; report-uri https://twitter.com/i/csp_report?a=05RXE===&ro=false
```

Summary

Try not to focus on the payload, but instead think of the
"attack chain"

Look for opportunities to break the attack chain and break it
in as many places as possible.

Building and Breaking: Web Applications

Holly Grace Williams



Thank you for listening!

If you've got feedback or questions, fire them to:

info@AkimboCore.com

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