Attack Atlas

sddec-21-16

Website for Sharing Threat Information

Advisor/Sponsor:
Lotfi ben Othmane

Members:
Jacob Abkes
Dylan Black
Andy Dugan
Jack Phillips
Zhi Wang

Problem

- Computer usage increasing day-by-day
- Ensuring these systems are secure is becoming increasingly important
- Attempting to secure computer systems reliably requires knowledge of many possible attack vectors

Solution

- Web Application
  - Single source of truth for attack vectors
  - Database of threat modelling patterns
  - Information provided by verified security experts
  - All easily searchable
  - Includes visualized statistics for threat models

Context

Threat Modelling - Threat modeling is a process by which potential threats, such as structural vulnerabilities or the absence of appropriate safeguards, can be identified, enumerated, and mitigations can be prioritized.

Threat Modelling Patterns - As defined by our project, this is a form of generalizing threats in order to create a hierarchy amongst risks including linking solutions, causes, and mitigation strategies in an effort to centralize threat related information.

How it started

1. Wordpress for blog posts

2. Tested with Angular and Spring Boot

3. Change in implementation and elicitation of requirements

Changes over development

1. Wordpress didn’t provide all the features we needed, we scrapped Wordpress and built blogging system in React and Node.js.

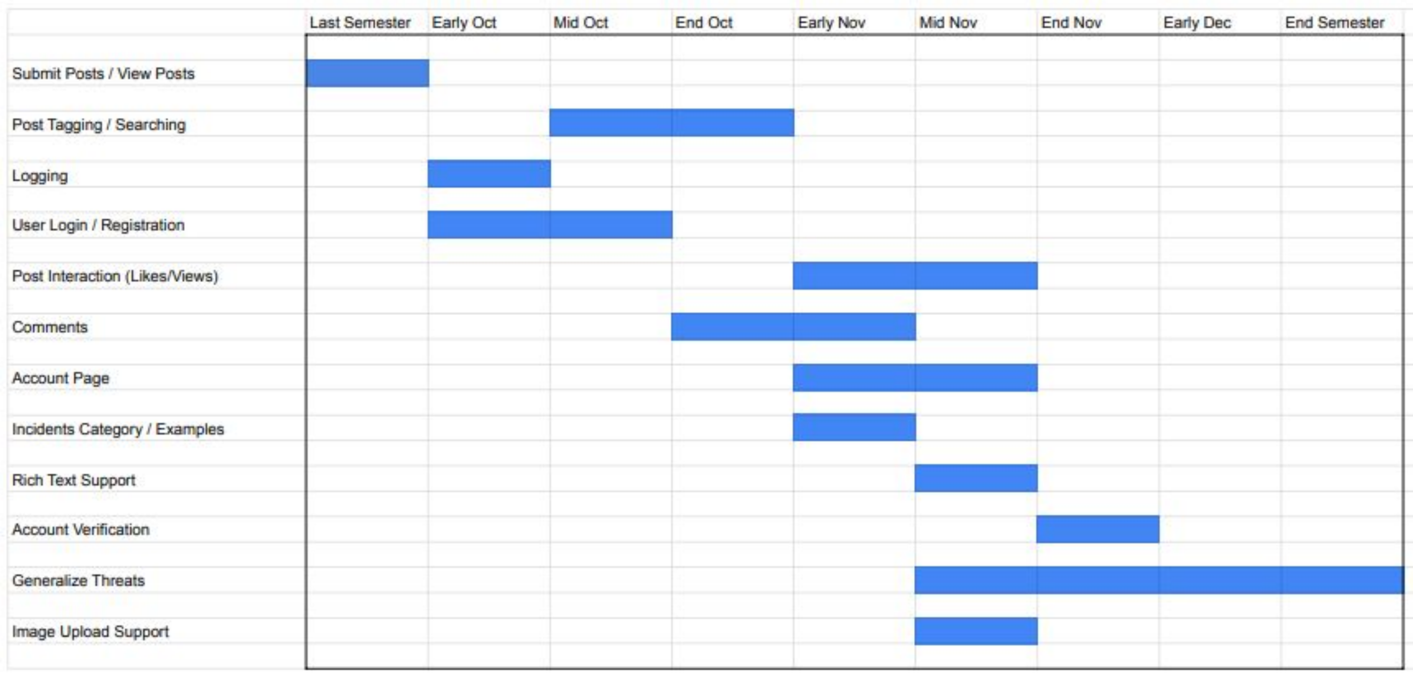
2. Before deciding on React and Node.js, we experimented with Angular and Spring Boot as front-end and back-end options, but it introduced too much overhead.

3. Over the course of the first semester the team gathered and attempted to implement Business Requirements without understanding how to meet the client’s needs through design.

Feature list

- Account System
- Threat Model Submission w/ Rich Text Support
- Incident Example Attachment
- Community features
- Post search w/ Tabular Display

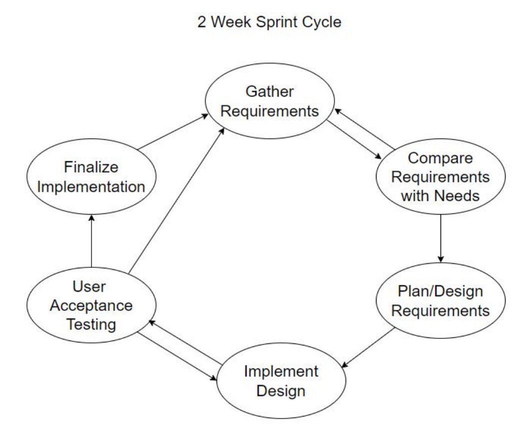
Retrospective Roadmap

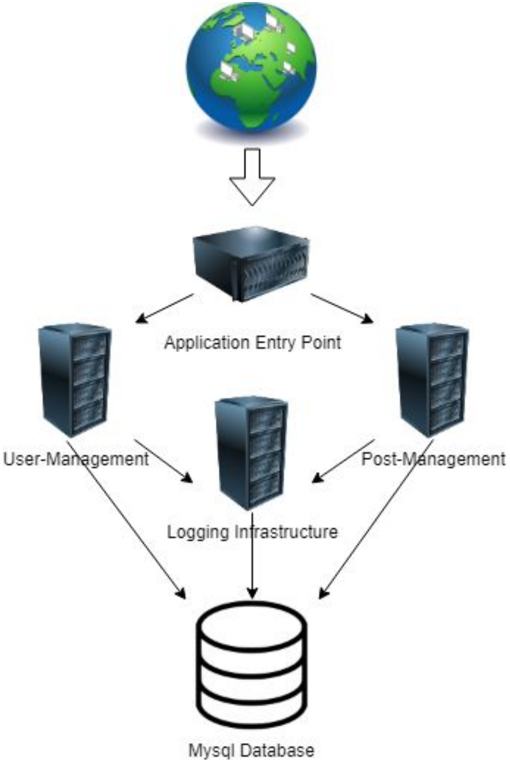


	Last Semester	Early Oct	Mid Oct	End Oct	Early Nov	Mid Nov	End Nov	Early Dec	End Semester
Submit Posts / View Posts	🟦								
Post Tagging / Searching			🟦	🟦					
Logging		🟦							
User Login / Registration		🟦	🟦						
Post Interaction (Likes/Views)					🟦	🟦			
Comments				🟦	🟦				
Account Page					🟦	🟦			
Incidents Category / Examples					🟦				
Rich Text Support						🟦			
Account Verification							🟦		
Generalize Threats						🟦	🟦	🟦	🟦
Image Upload Support						🟦			

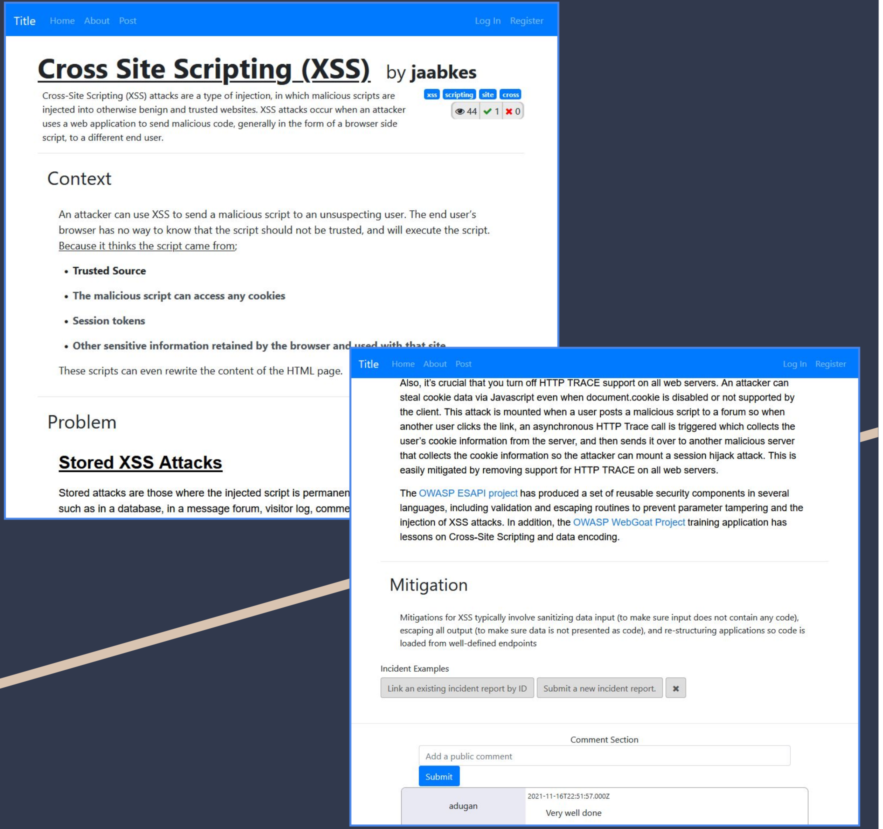
Demo

Design Process





Front-End



Cross Site Scripting (XSS) by jaabkes

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user.

Context

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from;

- **Trusted Source**
- **The malicious script can access any cookies**
- **Session tokens**
- **Other sensitive information retained by the browser and used with that site**

These scripts can even rewrite the content of the HTML page.

Problem

Stored XSS Attacks

Stored attacks are those where the injected script is permanen
such as in a database, in a message forum, visitor log, comme

Also, it's crucial that you turn off HTTP TRACE support on all web servers. An attacker can steal cookie data via Javascript even when document.cookie is disabled or not supported by the client. This attack is mounted when a user posts a malicious script to a forum so when another user clicks the link, an asynchronous HTTP Trace call is triggered which collects the user's cookie information from the server, and then sends it over to another malicious server that collects the cookie information so the attacker can mount a session hijack attack. This is easily mitigated by removing support for HTTP TRACE on all web servers.

The OWASP ESAPI project has produced a set of reusable security components in several languages, including validation and escaping routines to prevent parameter tampering and the injection of XSS attacks. In addition, the OWASP WebGoat Project training application has lessons on Cross-Site Scripting and data encoding.

Mitigation

Mitigations for XSS typically involve sanitizing data input (to make sure input does not contain any code), escaping all output (to make sure data is not presented as code), and re-structuring applications so code is loaded from well-defined endpoints

Incident Examples

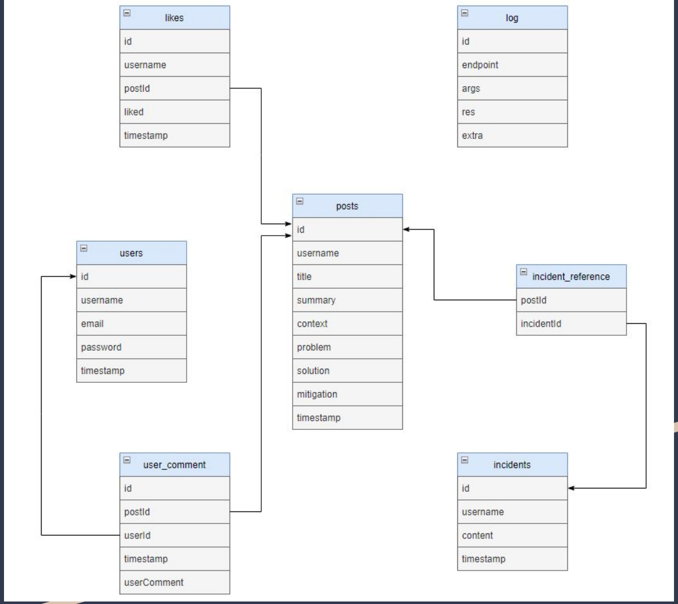
Link an existing incident report by ID
Submit a new incident report.
✖

Comment Section

adugan
2021-11-16T22:51:57.000Z
Very well done

- Based on React
  - React-bootstrap
  - Draft.js
  - Custom CSS where needed
- Utilizes jQuery and AJAX for communication with the back-end
- React is a javascript framework that allows for fast development by providing:
  - Access to 3rd party libraries
  - Fast data binding
  - Ability to create components
  - Easy to learn with prior knowledge of JavaScript

Back-End



likes
- id
- username
- postId
- liked
- timestamp

log
- id
- endpoint
- args
- res
- extra

posts
- id
- username
- title
- summary
- context
- problem
- solution
- mitigation
- timestamp

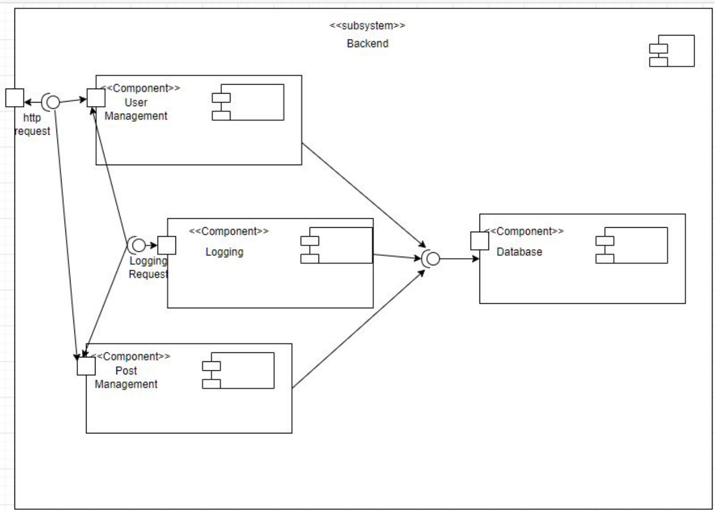
users
- id
- username
- email
- password
- timestamp

incident_reference
- postId
- incidentId

user_comment
- id
- postId
- userId
- timestamp
- userComment

incidents
- id
- username
- content
- timestamp

- MySql Database
- NGINX
- Node.JS Runtime Environment
  - Express.JS API Framework
- Runs on a virtual machine
- Postman



Conclusion

What went right

- Development
- Workflow
- Architecture
- Technologies

What we would do differently

- Understanding client needs
- CI/CD
- Automated UI Testing