Fake Stack Overflow Threat Modeling

Authors: John Jacoby, Yue Wen Peter Li
Course: CS5500 - Foundations of Software Engineering
Term: Spring 2024
Professor: Joydeep Mitra

Five Possible Threats

Threat 1 - Spoofing of User Credentials

Many of our requirements from the requirements engineering assignment involve users needing to be authenticated before performing an action, such as creating and commenting on posts. Without proper safeguards, an attacker could spoof the user's session and get the server to perform actions on that user's behalf that they did not actually initiate. For instance, the attacker could leave a comment on a post that contains HTML code and a script tag. When another browser then loads that comment, the code is interpreted by the browser, allowing the attacker to run arbitrary code on another user's browser. The script could, for instance, access the user's cookie with document.cookie and then send it to the attacker so that the attacker can make requests to the server using the credentials of the target user.

Threat 2 - Elevation of Privilege to Moderator Status

Requirement #8 defines a moderation system where specially privileged users called 'Moderators' can perform extra actions that other users cannot, such as editing, removing, flagging, or locking posts that are not their own. This obviously gives these users extensive power to alter and/or damage website content. An attacker could perform an XSS attack as described in threat 1 and hope to get a moderator's session. If the session IDs are not managed securely, an attacker could also trick a moderator into logging in with a pre-set session ID, for instance by sending a phishing email that tricks them into logging in with a link that already has the session ID set as a query parameter in the URL. Then the attacker just uses the same session ID to send requests to the server with moderator privilege.

Threat 3 - Tampering via Database Code Injection

All of our requirements implicitly require a persistent data store running on the backend to manage data storage and relationships for things like users, posts, and comments. An attacker can enter database code into fields like 'username' or 'post title' and when that code hits the database, it runs it instead of interpreting it as part of the input string. This could theoretically allow an attacker to do anything they want with our database - change user permissions, change user login info, edit existing posts and comments, or even delete the entire database.

Threat 4 - Denial of Service

An attack could write a script that just makes login request after login request (or any other kind of request) and takes up all of the server's cycles or all of the database's cycles so that it doesn't have any resources to respond to legitimate requests, and the site appears 'frozen' or 'down' to every other user.

Threat 5 - Information Disclosure via Man-in-the-Middle Attack

If login information is sent in plain text from the client to the server, an attacker listening to communication between the client and the server could simply read the user's login information.

Threat Specification

Threat 1 - Spoofing of User Credentials

All defined cookies must set the httpOnly flag to true. All user input must be sanitized by encoding HTML characters with their entity names/numbers. It should be noted that we can't entirely reject input that contains HTML characters because it is reasonable that our users will want to ask questions involving snippets of HTML code.

Threat 2 - Elevation of Privilege to Moderator Status

All defined cookies must set the httpOnly flag to true. All user input must be sanitized by encoding HTML characters with their entity names/numbers. Session IDs must be randomly generated using an accepted standard for cryptographically secure random generations, i.e. Node.js' crypto.randomBytes(length).toString('hex'), where 'length' must be at least 32 bytes. Session IDs must be re-generated upon every new login.

Threat 3 - Tampering via Database Code Injection

Mongoose schemas will be used to define exactly what the structure of a document should be. Mongoose parameterized queries will be used exclusively instead of direct database access. It should be noted that we can't entirely reject input that contains MongoDB symbols or JSONs because it is reasonable that our users will want to ask questions involving snippets of MongoDB code.

Threat 4 - Denial of Service

Session metadata will be calculated and stored. Every 30 seconds a '# request' field will be re-calculated for each session that determines the number of requests that that session has made in the last 30 seconds. If it's over 100 requests, the session will be considered invalid and closed.

Threat 5 - Information Disclosure via Man-in-the-Middle Attack

A standard and popular public-private key encryption implementation will be used to encrypt password data as it is sent from the client to the server.

Three Top-Priority Threats to Address

Threat 2 - An unauthorized user with moderator privileges could wreack havoc on any target user's post by deleting it or changing it to say anything they'd like, including offensive or illegal material that violates the site's policies. This is also a 'many birds with one stone' threat to address as XSS attacks can come in a variety of flavors that could all be damaging to target users in different ways. For instance, an attacker could also redirect a user to a phishing site that steals their login credentials. We also solve Threat 1 by solving this threat.

Threat 3 - Tampering via Database Code Injection

This is the most vital to address because the failure to address this threat is the most costly. An attacker could delete the entire website's database or give any user permission to perform any action.

Threat 5 - Information Disclosure via Man-in-the-Middle Attack

Sending passwords in plain text over the internet is extremely exposing. It's an exceptionally low bar for even amateur attackers to achieve, meaning that this will be one of the most common and often-attempted attacks out there. Users losing their accounts all the time to hackers will create a terrible user experience and make people not want to use the site.

Final List of Requirements