CPSC 329 - Group 40

Richard Gingrich
Kareem Abdelaty
Andrew Barnett
Ritwika Neupane
Brayden Schmaltz-Campbell

Introduction

We propose an interactive website showcasing different types of attacks. We would like to have the user interact with the field normally by inputting login information and then send a message telling them how their information is being compromised. We also plan on providing details about each type of attack, such as how it works and a brief summary of its background. In addition to this, we want to include some facts about how the attack has been used in the past and how it may be prevented.

Methodology

We intend to build the website using the Python3 Django framework and a basic SQL database built using the MySQL Library. We want to host the website on the university's CPSC servers. If the CPSC servers are not available then we will use AWS. We will be using GitHub as a source control software. All users will be informed when they first visit the website that the website is unsecure and as such they shouldn't enter real information.

Some examples of attacks we might use are discussed below.

The first type of Attack that we could showcase is the SQL injection attack. A SQL injection attack is when an attacker tries to submit partial or full SQL queries to an HTML form hoping that the query reaches the backend SQL database where the database would process the request and leak the information to the attacker. Although a SQL attack is very simple to attempt it could lead to disastrous consequences where the attacker ends up gaining access to the full database.

Cross-site scripting attacks are the most common types of attacks attempted on websites where they represent 40% of all attacks according to Precise Security. Cross-Site attacks actually target the user's browser instead of targeting the user themselves. They work by tricking the browser into running a script that is not actually part of the original website which allows the attacker to modify the website content to show the user what they want or even redirect the user to the attacker's website.

Since cookies are stored on local machines and not on secure servers, any user is able to edit their own. For example, if a website uses cookies to check if an account has moderator privileges, an attacker could easily trick this website. This leads to unwanted users having access to more on the website than they should. We intend to showcase this by having our website check if a user has the ability to edit the passwords file through cookies.

Roles & Responsibilities

Ritwika Neupane	Frontend Developer
Kareem Abdelaty	Backend Developer
Richard Gingrich	Backend Developer
Andrew Barnett	Documentor
Brayden Schmaltz-Campbell	Flex

Timeline

Feb. 7, 11:59pm: Rubric and Proposal Submitted
Feb. 8 - 12: Research & Learning APIs
Feb. 13 - Mar 12: Development
Mar. 12 - 15: Final Tests
Mar. 15 - Submission

Rubric

Please see the “Rubric.pdf” file.