

Websys Final Project Report:

HACK&/

By:

Finnegan Pike - pikef@rpi.edu

Arron Look - looka@rpi.edu

Sebastian Castillo-Sanchez - castis2@rpi.edu

Wilson Wong - wongw6@rpi.edu

Section: 2

Course Instructor: Dr. Thilanka Munasinghe

Grad TA: Satej Sawant, TA: Prem Datre

Due Date: December 11, 2018

Table of Contents

| <u>SECTION:</u> | <u>Page:</u> |
|---------------------------------|---------------------|
| Introduction | 3 |
| Problem | 3 |
| Proposed Solution | 3 |
| Users | 3 |
| Site Map | 4 |
| Features/Functionality | 4-7 |
| Goals Fulfilled | 7-8 |
| Future Goals | 8 |
| Individual Contributions | |
| 9-11 | |
| What We Learned | |
| 11-12 | |
| Problems Overcome | 12-13 |
| Website Reusability | 13 |
| Conclusion | 14 |
| Appendix | 14-23 |
| 1. <i>Github Link</i> | <i>14</i> |
| 2. <i>Screenshots</i> | <i>14-23</i> |

Introduction

HACK&/ has created a website that helps RPI students and people to learn about cybersecurity topics. This website will allow teachers or “admins” to create intuitive and interactive tutorials/challenges to give users a strong learning experience. The topics in this website can range from: learning Web Exploitation, using the Command Line, and providing users a well-rounded idea on how these topics influence the field of cybersecurity. Tutorials will be focused towards learning the skills, while challenges will test the user’s knowledge of the skills learned through the tutorials.

Problem

RPI students who are in clubs, such as INTROSEC and RPISEC, have a problem where they want to learn about the topics the clubs teach, but in a slower pace and an intuitive style. Most of the time, the club’s teachings are for more experienced and knowledgeable people that understand the details of each cybersecurity topic. As a group, we were experiencing a lack of knowledge in this field, so we wanted to create a solution that would help us with our troubles.

Proposed Solution

Our solution was to create a full-fledged website that will change the learning experience of anyone who wants to know more about cybersecurity. Our website gives “admins” the ability to create tutorials and challenges; that will give users the chance to improve upon their own personal skills. The users will be able to complete the posted tutorials as many times as they would like and complete challenges, until the admin decides to remove the content they created. The users would be able to see other user ranks through a scoreboard. This scoreboard ranks users according to how many points they have from completing challenges, each worth a hundred points.

Users

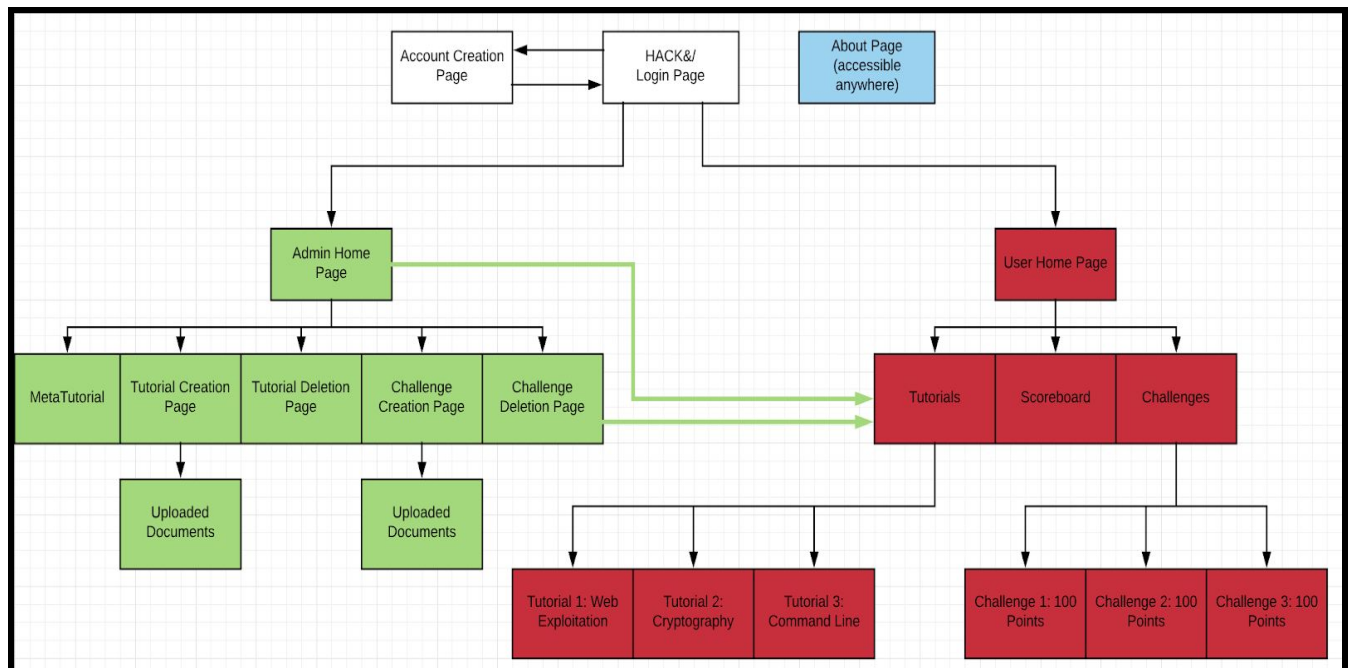
Our group wanted to reach out to students and people who want to learn about cybersecurity topics in an interactive way, and on their own time. They can gain progress in their knowledge by completing tutorials and challenges on general security or CTF topics. People who are striving to become white hat hackers can use this website to sharpen their skills and continue on their journey, learning all things related to cybersecurity. In addition, this website can be of use for people and students who want to meet other aspiring cybersecurity professionals. Overall, this website is for people who want to become enriched in the cybersecurity field regardless of prior experience and knowledge.

Site Map

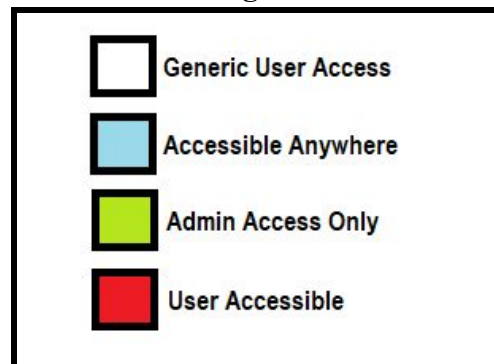
*Admins have the ability to view User pages in addition to Admin pages, but the Users can't see the Admin pages.

*The About Page can be accessed anywhere in the site.

*The tutorials and challenges in this site map serve as examples and are not actually implemented



Legend



Functionality/Features

The website has many functions that work towards helping people learn about cybersecurity. Tutorials and challenges have to be created and deleted, providing content for

users to interact with. Users must be able to complete challenges and tutorials to help them gather knowledge about and practice cybersecurity skills. The website itself must also be secure and have admin and normal user authorizations.

The registration PHP file is used for new user registration and only creates regular users, not admin type users. Admin users can only be created by the site administrator. All fields in the form on the registration page are required, with sanity checks on the backend once the user has submitted information. This is to ensure that the fields are completed and match certain specifications. The specifications are: entering the same password twice, checking that the password is at least twelve characters long, entering a valid email, checking that the username and email do not already exist in any entry on our “users” table, and checking that the username is under twenty characters long. Successful registrations results in the user’s credentials being stored on our “users” table, along with a randomly generated salt and their hashed password using the sha256 algorithm. Finally, in order to add an admin user, a normal user account must be made and the site administrator must include that user’s id in the “admins” table.

The index.php file in the root directory contains PHP code for user authentication and authorization. It contains a form that posts the username and password entered by the user to the same PHP file, that checks if the user credentials match their entry in the “users” table in our MySQL database. Upon successful validation, the user would be redirected to their respective homepage based on their authorization. For both the index and registration PHP pages, if the user is already logged in from a previous session the site would redirect them to their homepage. The sessions would store the user’s username and their authorizations, which is either a registered user or admin. This is used to check on each admin required page, such as the tutorial and challenge creation and deletion pages. If the user’s authorization is not “admin” on an admin required page, there would be a redirected to a normal user homepage. There is also an authorization check on the user homepage and admin homepage, so if the normal user accessed the wrong homepage, it would redirect them to the right page. At the top of every PHP file, there is code to ensure that the user is logged in to a session, otherwise the page would redirect to the login page. After all the overhead, the site user is guaranteed to be on the appropriate page based on their credentials, so only admins would be able to access the tutorial creation and deletion page as well as the challenge creation and deletion page. When the challenge and tutorial deletion pages are loaded, the SQL query only returns and displays the tutorials or challenges that specific admin created. That way, each respective admin would not be able to delete another admin’s tutorials or challenges. All registered users are granted access to the tutorials landing page, challenge landing page, and scoreboard page. The tutorial landing page contains all the tutorials the registered user completed or have yet to complete. The challenge landing page contains all challenges the user has or has not completed. The scoreboard page is a read only page that displays rankings and the user’s specific score. Overall, the users and admins have read access to the tutorials, challenges, and scoreboard where the content on the page is updated based on what the user has clicked or submitted. Only admins can create and delete their own tutorials and challenges.

HACK&/ has pivoted from its original goal of using PDFs in order to provide learning material for the new hackers. Instead we have given the admin the ability to upload an HTML document, which gives the admin dynamic control of how they want their tutorials to work. The HTMLs the admin uploads are displayed in an iframe in order to provide a degree of separation

from their code and the website. We also ensure the iframes are not editable so that the admin is not able to change the contents in the iframe to become malicious. However, one of our concerns was the fact that we wanted layman, people who don't know HTML or programming, to be able to create their own tutorials as admins too. We found a way to overcome this, since a layperson could create a Word Document and then create HTML from that. Any person can easily use the Microsoft Word with its simple functionality, and from this we provide the admins instructions on how to save their Word Tutorial as an HTML. This way we compromise between the two groups: the programmers that want to make dynamic tutorials and the laymen who have no idea how to code. This allows both groups to easily create tutorials.

One more functionality that we created in order to make the website more extensive for a layperson to use was to add in a question creator. The coding admin can create their own scripts, and if they want to add other functionality to their HTML pages they can add CSS and JavaScript to embed in their code. However, the layperson doesn't have this knowledge, so in order to improve the tutorials, we created a question creator. We allow the admin to create short answer and multiple choice questions. They can even see how the question would function in their tutorial. For the short answer question the user has to input the correct answer, or click the "Show Answer" button to show the answer. For the multiple choice question the user has to click the correct multiple choice answer for the answer to be shown. This does make it easy for the user to find the answer if they really do not want to think about how to answer the questions, but our website is not focused on making it hard for the user to learn. If the user wants to obtain the value of learning then they will think about the questions, instead of just clicking "Show Answer" or all of the multiple choice answers. We allow the user to determine how they want to learn, and what they will get out of the website is whatever effort they put forth. This also allows the admin with little coding experience to add variety to their tutorials.

However, we did add some restrictions to the tutorial creation. The multiple choice questions can have only two to five answers to make it so the admins can not create a question with too many answers. We also made sure that the user can only have at most twenty pages per tutorial. We don't want admins to create tutorials that are too long because lengthy tutorials could not only bore a user, but also bog the website down with too many files. However, the tutorials can be long enough so that the admin can create an in depth tutorial with a variety of questions and tutorial pages. The admin also can not upload any file that is too large to load down the website, that creates an error, or is not an HTML page. We ensure that the file extension is appropriate for the sanity of the website. We also ensure all of the inputs are filled out in order for an admin to create a question.

After the admin has uploaded all the documents for their tutorial, and have created any of their questions, they can scroll down in the tutorial creation page and see what they have already uploaded. Each page of the tutorial has its own box which the admin can click to preview and see the entire contents of their HTML. If the admin wants to delete a page from the tutorial they are creating, they would press the delete button in that page's box. Once all the pages have been successfully chosen, with deletions of any unwanted pages, the admin can then drag the pages to create the order they want, and then submit the tutorial with a tutorial title. The tutorial is then created in the tutorial page for all the users to complete. If the admin wants, they can delete this tutorial if they do not like it, and recreate it in the tutorial creation page. However, this is not the only action the admins can take in providing content for users.

On the challenge creation page, admins are able to provide a challenge name, a challenge description, a flag for the challenge, and at most one file pertaining to the challenge. All of this data is taken in from a form so that the challenges can be uploaded by a layman. Challenge files are currently limited to HTML files, text files, and SQL files, due to the security issue posed by allowing content creators to provide certain files, such as PHP. The plan is to have the admin submit a request to the site administrator, with all the files the challenge would require for a quick screening to check for security issues in the challenge creator's code. Each challenge created stores, at most, two files on the server. By default there will be one file, a text file, created that would contain the content to show the user when they first select a challenge to view or solve. If the administrator uploaded a file for that challenge, that file will also be stored on a separate location on the server instead of the same folder as the text file.

On the challenge landing page, users are able to view and interact with challenges they have yet to complete. For each challenge the user successfully completes, they gain 100 points on their overall score. Once a challenge is completed, there is no way for users to reuse that challenge to continuously gain points for each resubmission. Similarly, the tutorials landing page displays links to complete and incomplete tutorials. At the end of each tutorial, clicking the "END" button will mark that tutorial as complete for the user and return them to the tutorial landing page where they can view which tutorials are finished. All registered users are able to access and re-access any existing tutorials regardless of their completion status on their screens.

For admins, the challenges and tutorial deletion page operate on similar principles where the admin is only able to delete tutorials or challenges they made. The admin simply needs to click the "DELETE" button for the challenge or tutorial, confirm their action, and every user's challenge or tutorial bitstring -- depending on whether the admin is deleting a challenge or tutorial -- is updated to exclude the bit information about that tutorial or challenge. If the admin deleted a challenge, all user scores are updated to reflect the removal of that challenge as if they never earned that 100 points, due to the unfair advantage that veteran users of our site have in the sheer amount of challenges they have completed overtime. That way, all users have access to the same number of challenges and points providing a fair and unbiased perspective on a user's skills compared to other users, based on their ranking.

We wanted the website to have a horror theme for multiple reasons. Hackers generally like to work in night mode, since night mode makes screens easier to look at. We also wanted to have a fun theme, which the corky horror brand does provide. Lastly the horror theme goes with the HACK and SLASH brand, and continues to have many other correlations with hacking like: John the Ripper password cracker, Jason Voorhees from "Friday the 13th", viruses, black hat hackers, etc. We thought it would make for a more enjoyable website if there was a fun theme that comes with it.

Goals Fulfilled

We originally wanted our site to be one where admins would upload tutorials and users could view them at their own leisure. Later on, we developed and added the idea of having challenges in order to test a user's knowledge, as simply reading a tutorial isn't enough. Along with that, we decided to add minor questions to the tutorials to help the users along with the learning process. We also wanted to implement a scoreboard to rank users on our site to

encourage people to complete challenges to become the best hacker on the site. The scoreboard also serves as a reference for the user to gauge their knowledge in comparison to other users.

At first, we were planning to have admins upload PDFs to the website and display them. However, we decided to change to making a text editor so that anyone could create their tutorial in the website. We would then take the HTML and have it uploaded as one of their pages in the tutorial. However, a text editor is a finicky and complex project in itself, which led us to decide that working on it any further would be a detriment to the project. That's when we decided to switch to an HTML upload since a layperson could still make HTML through Microsoft Word. This essentially allowed for the prior objective to be completed indirectly, while giving people with the knowledge of HTML the ability to make their tutorials more dynamic.

Future Goals:

For future goals regarding our project, we would have liked to add several features. One such feature would be providing users the chance to reset their password if they forget their old login information. This would be incorporated by adding a "Forgot Password" page that the user can submit their email so they can go through two-step verification to reset their password. When it comes to tutorials and challenges, we would like to have a few full fledged tutorials or challenges on the site so users can start learning immediately. Another feature would be to have the ability to upload multiple files on the challenge creation page. We would like to incorporate our own HTML text editor in the future for more accessibility and efficiency in creating tutorials and challenges.

On the more technical side, to disallow malicious code to enter our website, we would like to implement an operation in the file upload process that will check for malicious code (like PHP). This will prevent attackers from injecting code into "tutorial" files that can destroy our databases, delete important files, etc. We would also like to create an email system to submit certain files so that site administrators can sift through those submission requests. To make the website more social and interactive, a future implementation would be to create a discussion forum. It would allow users and admins to ask questions, give advice and share ideas about tutorials, challenges and cybersecurity as a whole.

As of now, the scoreboard only has one spot for each rank. Even if users have the same score they are sorted by their username. We would like to change the ranking format to allow for ties.

On the admin side, we would like to create an application form so users can apply for admin privilege(s). Including an interface for the site administrators to approve admins will expedite this process as well as provide a way to remove admins for posting malicious/inappropriate content. For admins and users who need to ask questions about the site's functionality or report problems, having a system to email the site administrator(s) will be very useful. Finally, as much as we want this website to be about cybersecurity, incorporating other organizations and clubs to expand the context of our website will allow for more diversity as well as more content. Overall, these goals are attainable and we as a team are very excited to keep implementing these goals to improve the website even more.

Individual Contributions

Finn:

In this project I was the team lead. I helped facilitate the timing and location of the meetings, while dividing the workload between the members. I created the file structure for the website which separated vital pages from one another so that there was a degree of separation between users and admins. There was also a resources page created so that all of the content pages of the website could be linked to extra CSS, JavaScript, and image files. This way we could reuse content for the website. I photoshopped the background for the website from a regular forest photo to convey a dark atmosphere. Then I designed and created the logo for the website with photoshop. I then created the generic CSS for the website and made its overarching theme.

Then I began to work on the text editor. I worked a few weeks on the text editor, which was supposed to allow even a lay person to just type into the website and style what they write like Microsoft Word. However, I soon saw that continuing to work on the online text editor would be detrimental to the overall functionality of the website, which is when I transitioned into making the tutorial creation page.

I started the tutorial creation page by developing the question making part for the admin. I started this by making the HTML/JavaScript so the admin could pick between the short answer question and the multiple choice question. Then I made it so that the admin could see a preview of how their question they created would look like and function. Next I made the tutorial upload page so that the admin could upload HTML for their tutorials. I had to ensure that when the admin refreshes the page a post of a new file to be uploaded would not re-upload a file. I secured the upload so that if the file was not the correct type, too large, or had an error in uploading that the admin would be notified of the type of error and would not upload the file. If the upload worked then a message would tell the admin that the file had been successfully uploaded. Then I made it so that the question could be successfully made into a question HTML that would be part of the admin's tutorial. I created a PHP script that would show all of the HTML files that the admin specifically had uploaded, and then made it so that the admin could delete any of the tutorial pages that they didn't want in the upload. I developed a dragging system so that the admin could drag one of the tutorial pages and reorder how they wanted their HTML to be displayed. I made each of the HTML pages of the tutorial viewable in a preview that would come up on screen if they clicked the particular page they wanted. This way an admin could see what the HTML contents they had were instead of just seeing the file names. Lastly I made it so that the admin could submit the tutorial with a name so that all of the html pages would generate a tutorial for all of the users to see. This generated tutorial would have the order the admin drags to create, not the uploaded order.

I worked on both the frontend and backend parts of the website, while organizing the team and writing assignments. I was able to cut through the fine details in the writing, while producing the code needed for the website to be functional. I was able to give code to my fellow teammates so that their jobs would be easier, and made the code understandable so that they could integrate other scripts into their code. This was my job as leader of the project.

Arron:

Initially started on some of the HTML rough drafts for the tutorial and challenge home page. From there, deciding on and creating the database structure for storing the tutorials and challenges was required to proceed, with the users table created as a byproduct to store user credentials for our website. Since the admin tutorial and challenge creation and deletion pages were not functional yet, I decided to work on the login and logout functionality of the website based off the HTML template Wilson created for the login page. Once that was complete, I started working on the general user registration page, where one could only register as a user and not an admin. When that was complete, it was time to fix the navbar links for all the site pages and to ensure that any registered user or admin would be routed to the correct page based on their authority. Utilizing the HTML snippets that Finnegan made to make our site's navbar, header, and footer look uniform, I inserted PHP code to generate HTML links for the site's logo to bring the user back to their home page, for each navbar element to route to the correct location on the website, and for the logout button to be correctly linked. Once all the linking was complete, it was time to convert the HTML templates for the tutorial landing page and challenge landing page into PHP files due to the dynamic content we planned to serve -- any number of tutorial or challenges to present to the user. PHP was used to query our database for the tutorials or challenges to display on the tutorial or challenge landing pages, which classifies the content returned from the query based on whether the user completed that tutorial or challenge. From there, the scoreboard HTML template had to be converted to PHP to show the correct rankings, so I made a new file for the scoreboard page to show the top 10 ranking users, the user's current rank, and the user's current score. At this point in time, the challenge creation page had yet to be created so I used some of Finnegan's existing format for file upload and his HTML form on the tutorial creation page to implement the challenge creation page. To implement this page's functionality, I created a separate PHP file that would take the data posted from the challenge creation page to store one to two files for each challenge submission, one file containing the content to show users when they select a challenge to view and a second file if the admin uploaded a file for that challenge. It was then time for a challenge deletion page, a follow up to the challenge creation page. After making the page's layout to dynamically display a list of all the challenges created by that admin paired with "DELETE" buttons, I created a separate PHP file that would update each user's challenge bitstrings and each user's score as necessary to reflect a state where that challenge never existed -- so the points earned for that challenge never occurred -- then removed that challenge's files from our server before proceeding to remove that entry from the `challenges` table. Finnegan's tutorial creation page was mostly complete at this point so I started his tutorial deletion page for him. The tutorial deletion page would have the same layout and lead to the same PHP file that handled challenge deletion and would execute similar code as when deleting a challenge. When deleting a tutorial, it would update the tutorial bitstrings of all users, proceed to removing the tutorial files and folder from the server, then finally remove the tutorial entry from our table. Overall, I worked on most of the backend for the project by implementing the login and logout feature, the tutorial and challenge landing page that required dynamic content utilizing user bit strings, the tutorial and challenge deletion pages, the

challenge creation page, the scoreboard page, the dynamic linking of the entire website, and ensuring page access by user authorization.

Sebastian:

For my individual contributions, I was able to work on various things that helped the website's structure and functionality. I helped some with the database creation to ensure that they were created correctly and that the structure would work for our specific needs. In the beginning I was able to contribute to the first HTML and CSS code structure that was created to link pages and styles together. I was able to create the HTML code for several important pages for the site such as the About and Metatutorial pages. The Metatutorial page consists of instructions on how to create/delete tutorials and challenges so it would be able to give admins a clearer and concise idea on how to create/delete tutorials and challenges. I strongly focused on adding more user experience and usability to the website to ensure fluid using for admins and users. This was done using knowledge that we learned in class as well as side learning. On the tutorial side, I created a Web Exploitation tutorial that contained three different files exported from a word document to use for presenting and testing. During the end process of the project, I mainly focused my time on debugging/testing the website for various bugs and problems to ensure that a user can use the website efficiently in any browser or operating system.

Wilson:

I created many of the "starter" HTML pages, including the currently existing About Page and Metatutorial Page, in order to first have a basic file structure to begin working. I also taught the rest of the team how to use GitHub in order to streamline the process of sharing code with each other. I helped find many of the bugs in the PHP files as well as broken links, and helped test the functionality of the code by creating tutorials and challenges. I was also responsible for most of the existing CSS on the pages, barring the general CSS Finn made for all of the pages, and kept the file structure of the entire project organized. I began work on the tutorial file creation while Finn worked on his text editor. For testing, I made a separate folder with some basic files so as to not mess up others' work. I was able to get a version working for the client side, and started working on one that would work server-side. However, eventually we both gave up in order to implement a quicker method in order to finish the project in time. Lastly, I created the challenge which we used for the demo, which tested a user's ability to use MySQL injection. That consisted of a simple HTML "login" page with an unhidden password (since it is supposed to be more of a challenge, I made the password unhidden so a user could see what s/he was typing). That connected to a PHP file which displayed some credentials of a specific user. The goal was to get the credentials of every user on the database at once, showing off the dangers of MySQL injection.

What We Learned

While creating our HACK&/ project we learned about web development, project scheduling, project writing, and presentation skills. We learned how to safely log a person in, and to store their passwords with salting and the sha256 hashing algorithm. We were able to figure out how to use sessions, which helped us maintain the security of authorization between admins and generic users, while also allowing each admin to have the the ability to have their own folders to store their tutorial files. This way when each individual admin logs back into their

tutorial creation they can continue working on their tutorial safely. Sessions were a vital part of the project, and allowed us to implement many functions we could not have otherwise.

We explored the differences between browsers and found that Safari and Firefox are very similar to Google. There were minor errors between each browser, so we only had to spend a minimal amount of time dealing with those browsers. The major browser we did learn about was Internet Explorer, which does not have easy compatibility with other browsers. Many of the CSS elements that would look fine in Google, Safari, and Firefox would look hideous or not even have the same effects as in Internet Explorer. Internet Explorer also has problems with iframes, and we had to adapt the linking of certain iframes so that it had a src or srcdoc attribute. This finally allowed the project website to be compatible across the multiple platforms we planned to support.

We learned how to upload, delete, move, and generate new files. We learned how to use PHP to develop the backend of the website.

We understood that we would have to be consistent with meeting every week and getting work done, but we found out as the semester came along that we could have spent more time on the project. This way, when the semester became more difficult in the end, the project could have become less stressful and less rushed. We also learned that Slack was not the best way for us to communicate, since not every member used Slack enough to make it viable. Therefore we found that the best way for us to communicate with one another was group chat texting, since we all seem to check our phones for that more often.

While completing our project, we also found that our project writing was improving. We would look up how to professionally organize certain parts of our writing, and use that to better create project documentation.

Lastly, at the end of the project we had to learn how to present. We timed ourselves and decided who would present which part of the project. This allowed us to become better prepared for when we had to present for real.

Problems Overcome

When creating the login page, we had a little confusion over how websites stored user session information, but confronting our Grad TA for advice clarified our thoughts on the roles that cookies and sessions played.

The project specifications mentioned browser compatibility as a requirement and when checking for browser compatibility, we found Internet Explorer riddled with compatibility issues. Although Internet Explorer supports positioning, defining top, left, right, bottom values after setting position defines different behaviors on Internet Explorer, compared to Chrome and other browsers. There were other CSS issues with Internet Explorer that required the use of media query to provide different styles based on the browser. Internet Explorer also does not support the srcdoc attribute for iframes, so we had provided a src value for iframes if the browser was Internet Explorer whereas other browsers were able to use a srcdoc value. There was also an issue with Firefox not supporting “event.target”, but the work around for that was providing the event object when calling the function.

There ended up being major problems with the linking of some files. Specifically there were problems when there were certain characters in the file name. We ended up having to scrub and replace all characters that were apart of the NTFS windows file structure. Since Mac is less

strict with the characters in their file structure, we decided we would go with the Windows compatible file structure. This fixed all of our linking problems when creating tutorials.

We had quite a few problems with creating the text editor, which is why we switched to HTML uploads. One problem was that the text-editable content within iframes and divs had a weird structure anytime the backspace was pressed. This would delete *everything* in the iframe and div tag and would ruin any extra functionality of the text editor. Whenever an admin would press enter there would also be some weird results in the HTML that seemed to be unpredictable, and with time could probably be fixed, however it would have been detrimental to the overall functionality of the website. Therefore we switched to having the admin upload HTML files.

Website Reusability

All of the code is fairly commented with the purpose of each major segment stated. Future site administrators or developers who would like to reuse our code should be able to easily determine which parts they can reuse for their own project(s), and apply them while changing any specific details, such as the database or table names. The code snippets in `/resources/general/` could be modified and reused by developers who need headers for their webpages, footers for their webpages, a connection file to reuse every time they need to connect to their database, and more.

If a developer wanted to import the entire website to remodel for their own purpose, they can easily change a few links in the files located in `/resources/general/` since most of the links on the website are generated through PHP functions and the values in the PHP superglobals, `$_SERVER` and `$_SESSION`, so changing the file structure may alter the links which can be easily fixed because most of our PHP “require” statements reference some HTML or PHP snippet in the `/resources/general/` folder. Additional linking errors may occur because of the links in the “delete.php” and “upload.php” files in the `/admin/tutorial_creation_page/` folder, which are both used in creating a tutorial. So, it should be relatively simple to use our website files as a base. Our folder structure is self explanatory as well with each folder from the root containing a PHP file or more folders that each contain at least one PHP or HTML file. The resources folder from the root directory contains folders with JavaScript or CSS for each PHP page our website serves, making those resources easy to find and map.

For future site administrators that will be controlling the entire website, there is an “Installation Guide.txt” file containing instructions on how to install the website and access it through a virtual host run by Apache. For admin users, there is a meta tutorial they can always access from the site for them to learn how to create or delete tutorials or challenges. For registered users, there are only a few interface options making the website easy to grasp without a need for a site walkthrough. If there is still ambiguity as to how our website works, there is always the README files in each folder containing documentation about what that PHP or HTML file in the folder does for the site administrator to use as a reference.

Conclusion

In conclusion, team HACK&/ has successfully created a website that will give RPI students and people the chance to learn about cybersecurity topics in a very intuitive and interactive way. Throughout the process from start to finish, we worked very hard in completing our goals and it proved a success in the end. We are definitely very excited to see if we can implement the future goals mentioned above. This website will definitely create learning opportunities for people that have not had the chance to learn at their own pace and they will develop a learning experience that will help them in their future studies. HACK&/ will be a backdoor for users to have a fun learning experience at their own pace.

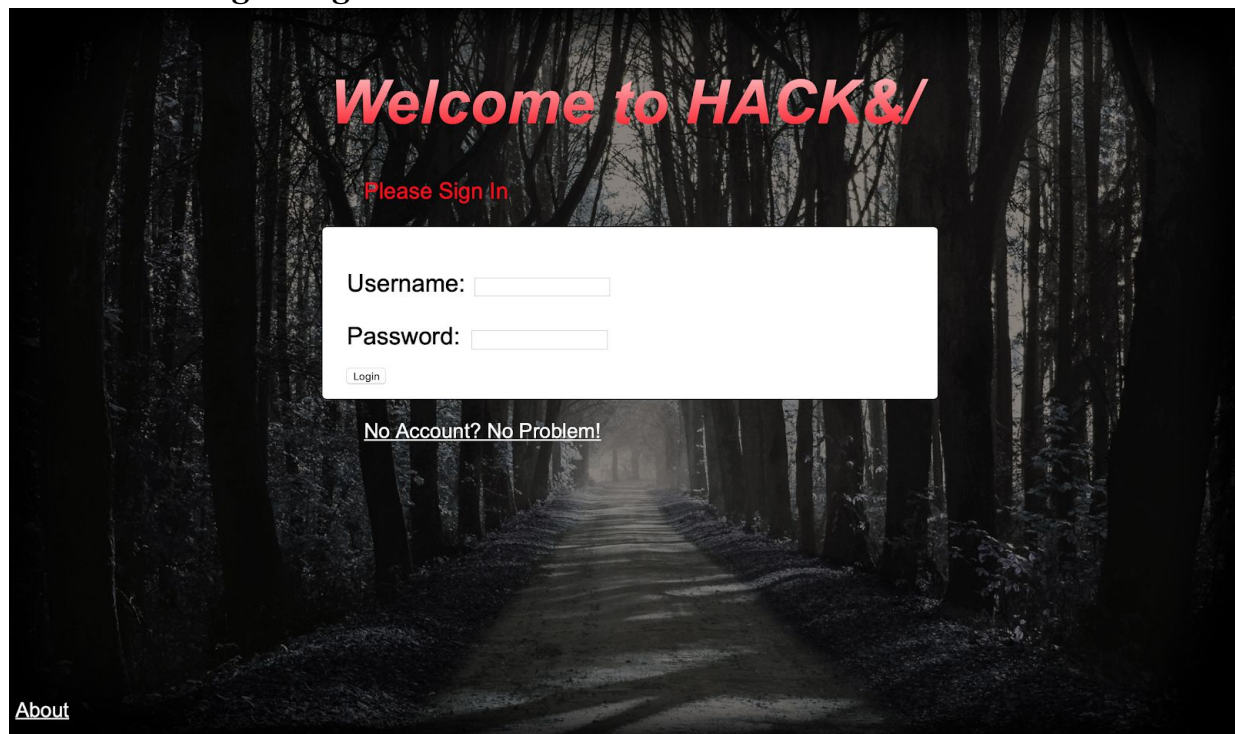
Appendix

1. Github Link

- a. https://github.com/estalcil134/HACK_AND_SLASH

2. Screenshots

a. *Login Page*



b. *Register Page*

Welcome to HACK&/

Please Create an Account

Enter an Email:

Create a Username: (under 20 characters)

Enter a Password: (at least 12 characters)

Re-enter Password:

[Already have an account? Login here!](#)

[About](#)

c. About Page

About HACK&/

Our Mission

HACK&/ is focused on creating a more efficient process to learn more about cybersecurity topics through interactive tutorials and lessons.

The Why

Rensselaer Polytechnic Institute has an prodigious club called RPISEC, which helps students learn about and utilize cybersecurity. The club is made of very experienced members who help other students get into white hat hacking. However, there is a level of experience students need in order to benefit and learn from RPISEC. Therefore, RPISEC made a secondary club, IntroSec, in order to lead inexperienced students into the cybersecurity world. RPISEC has made substantial progress in helping those novice hackers into the club, however there is still a steep learning curve.

It is hard to find the right tools, documentation, and tricks to start learning about cyber security, which is why HACK&/ is making a website to give these resources to students. HACK&/s objective is to give users a website where: they can learn about multiple cybersecurity topics in a tutorial, answer questions about what they learned to cement what they've read, and then practice their newfound skills in some challenges. The challenges will have scores so that users can see their progress in the challenges by seeing how many points they have on a scoreboard page. Admins will be the "teachers" of the website, who can create or delete tutorials and challenges, tailoring the website to create a better learning experience. HACK&/ will help people begin their journey of becoming experienced in the cybersecurity field.

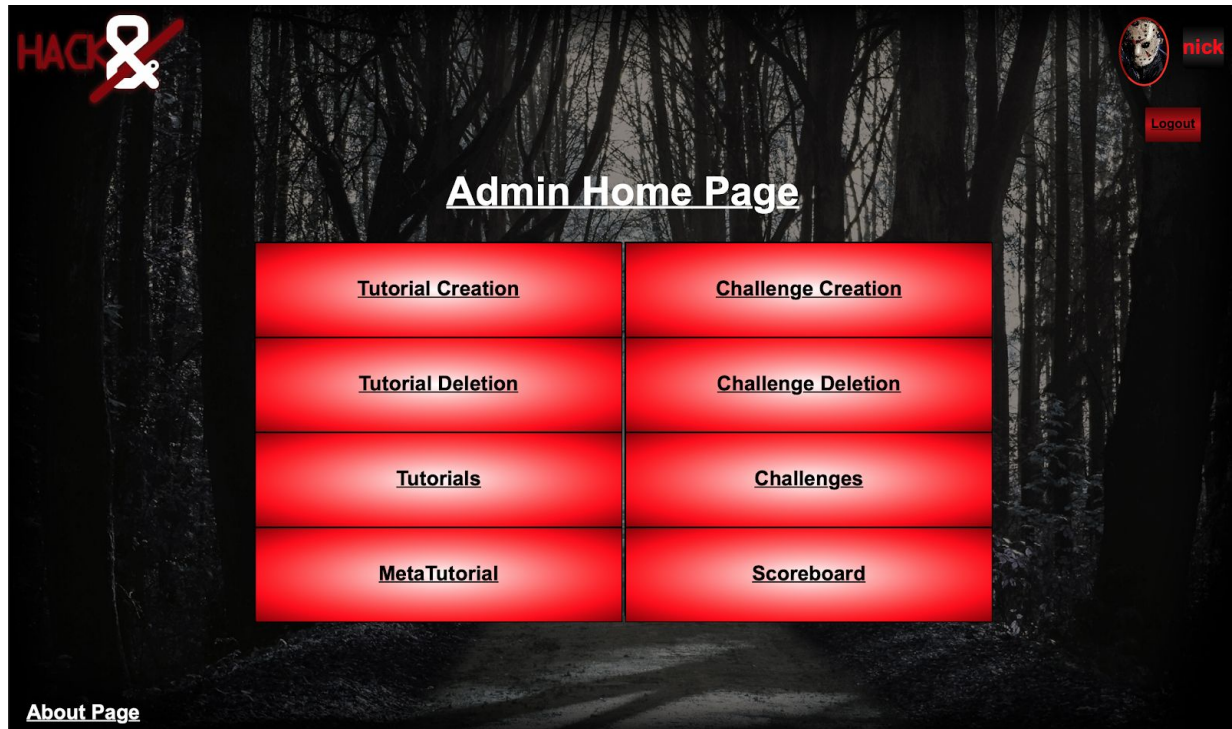
Who We Are



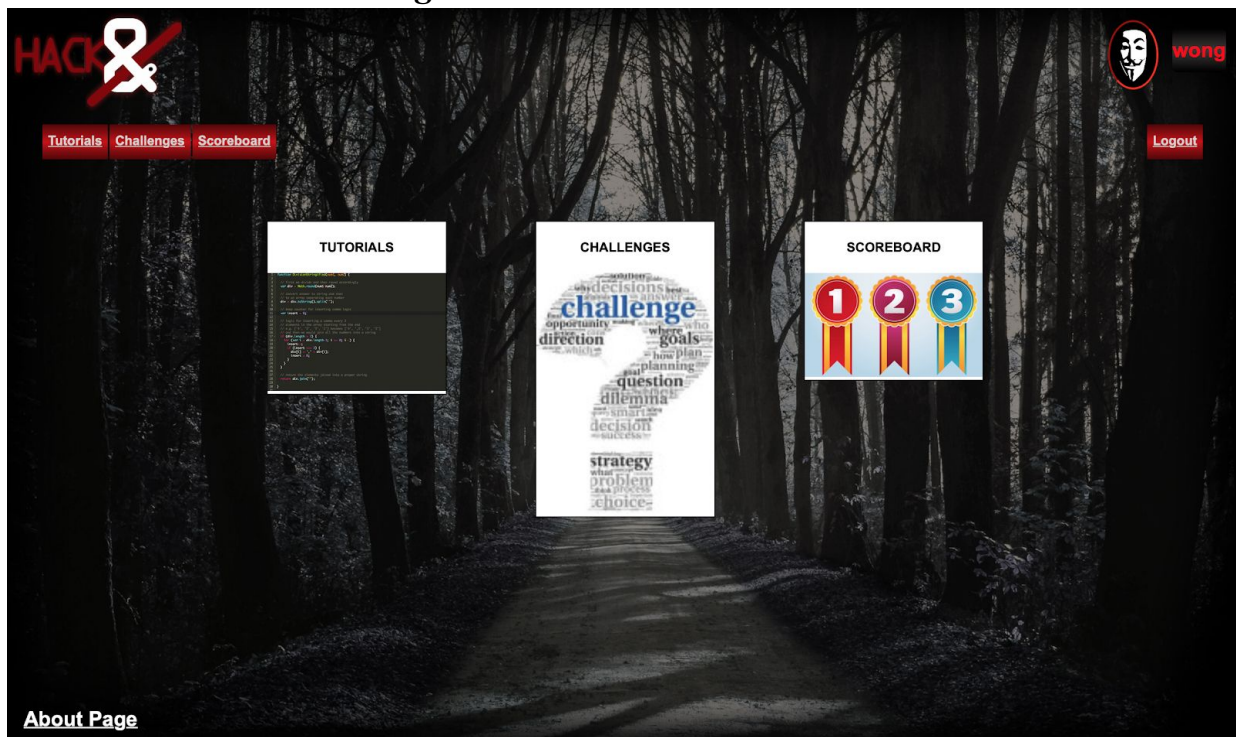
Finn

I'm an RPI student. I'm an Information Technology and Web Science and Computer

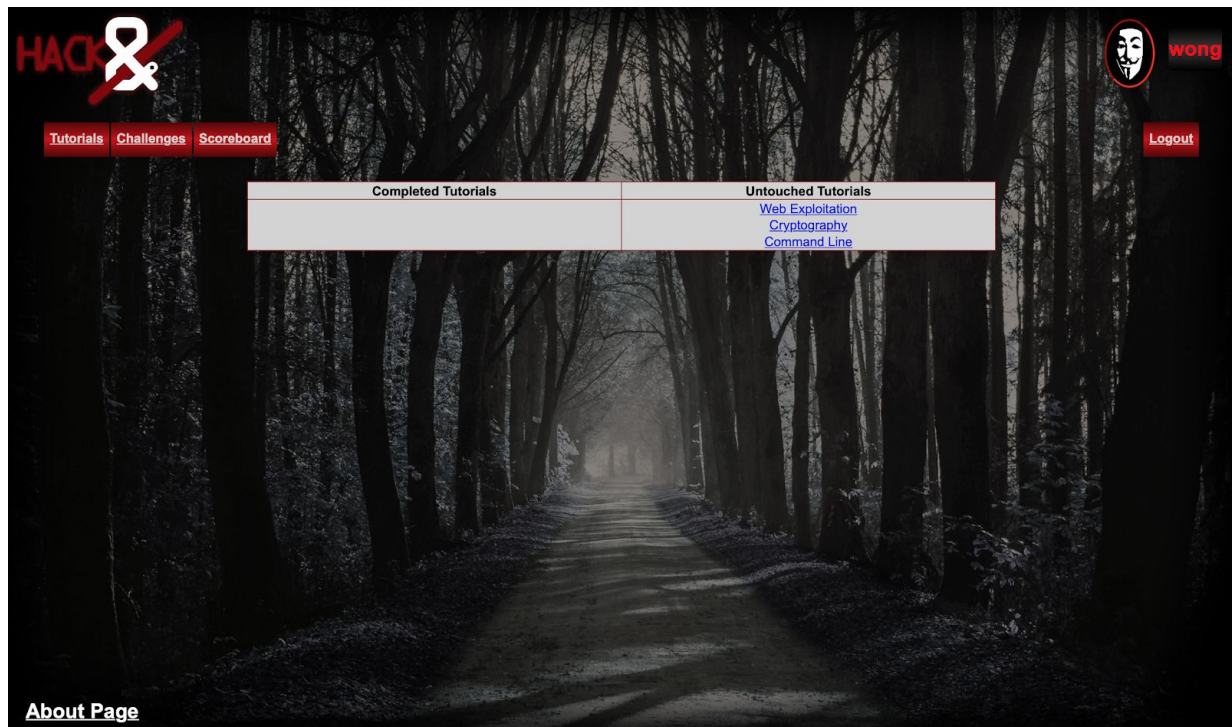
d. Admin Home Page



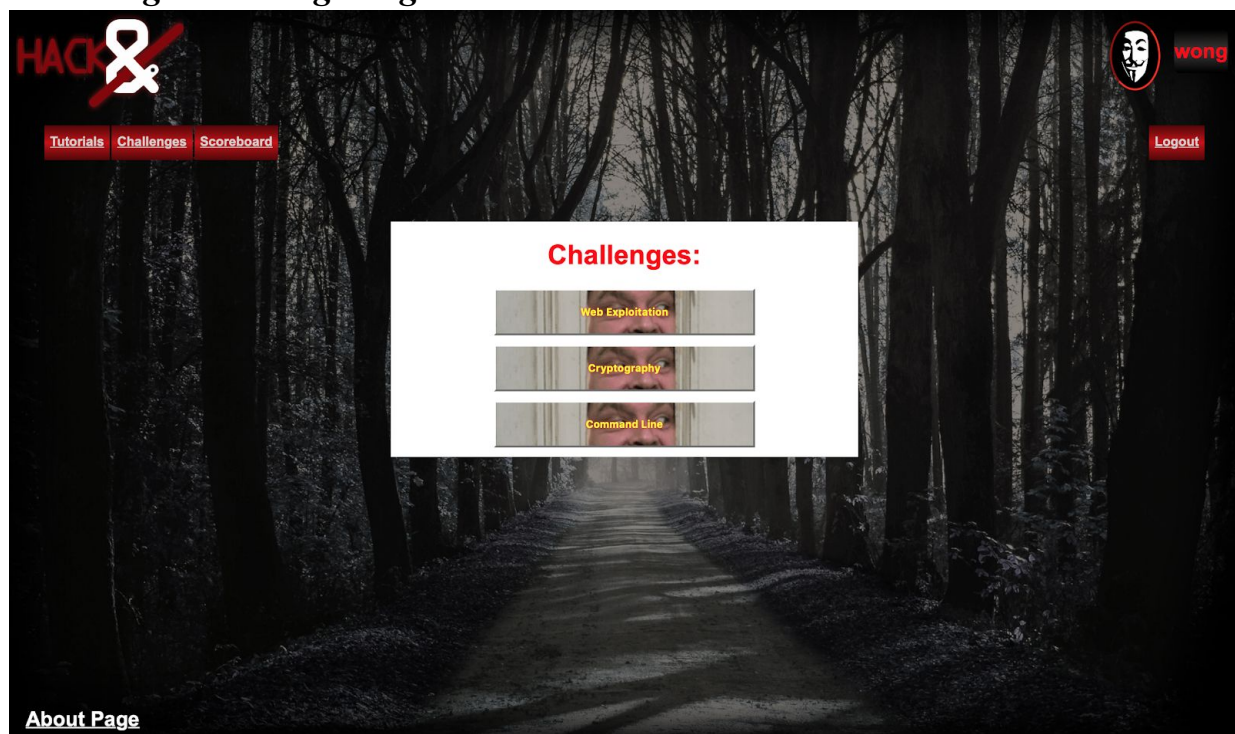
e. User Home Page



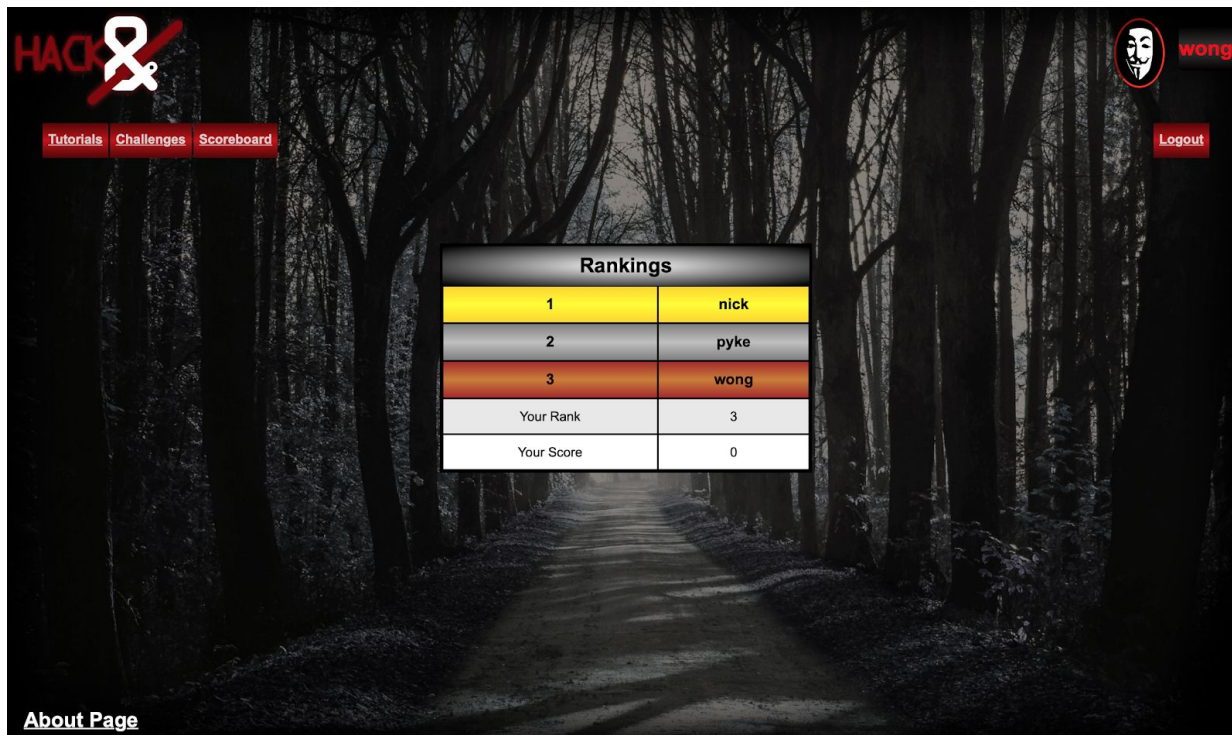
f. Tutorials Page



g. Challenge Page



h. Scoreboard

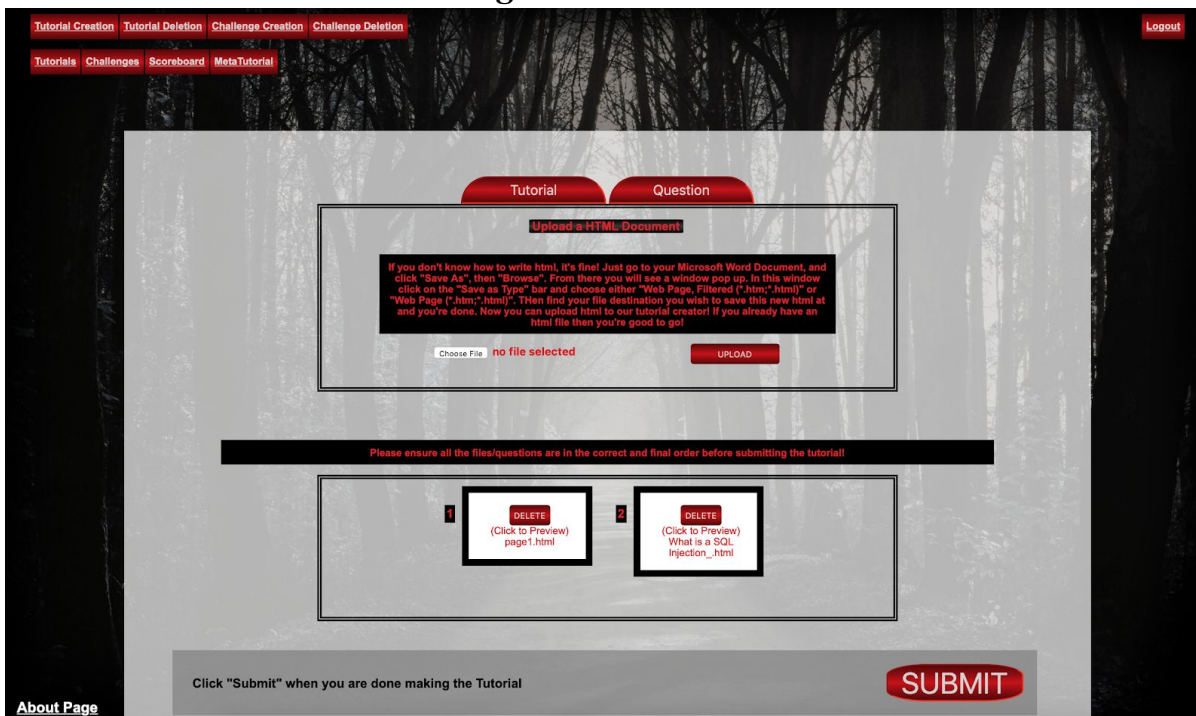


The screenshot shows the 'Rankings' page of the Hack & Wong website. The background is a dark, atmospheric image of a forest path. The top navigation bar includes 'Tutorials', 'Challenges', and 'Scoreboard' (highlighted in red). A user profile for 'wong' is visible in the top right corner. The 'Rankings' table is centered on the page, showing the top three users and the current user's status.

| Rankings | |
|------------|------|
| 1 | nick |
| 2 | pyke |
| 3 | wong |
| Your Rank | 3 |
| Your Score | 0 |

An 'About Page' link is located in the bottom left corner.

i. Tutorial Creation Page



The screenshot displays the 'Tutorial Creation' page. The top navigation bar includes 'Tutorial Creation' (highlighted in red), 'Tutorial Deletion', 'Challenge Creation', and 'Challenge Deletion'. A 'Logout' button is in the top right. Below the navigation bar, there are tabs for 'Tutorial' and 'Question'. The main content area is titled 'Upload a HTML Document' and contains instructions for creating an HTML file. Below the instructions, there is a file selection area showing 'no file selected' and an 'UPLOAD' button. A warning message states: 'Please ensure all the files/questions are in the correct and final order before submitting the tutorial!'. Below this, there are two preview cards for uploaded files: 'page1.html' and 'What is a SQL Injection.html'. Each card has a 'DELETE' button and a '(Click to Preview)' link. At the bottom, there is a 'SUBMIT' button and a note: 'Click "Submit" when you are done making the Tutorial'. An 'About Page' link is in the bottom left corner.

TutorialQuestion

To upload a question please enter the question title, specify type of question and input answer before uploading. There is a preview box down below for you to see how the question will appear!

Question:

Short AnswerMultiple Choice

Answer:

QUESTION:

SHOW ANSWER

CREATE QUESTION PAGE

Please ensure all the files/questions are in the correct and final order before submitting the tutorial!

1

DELETE

(Click to Preview)

page1.html

2

DELETE

(Click to Preview)

What is a SQL Injection_.html

Click "Submit" when you are done making the Tutorial

SUBMIT

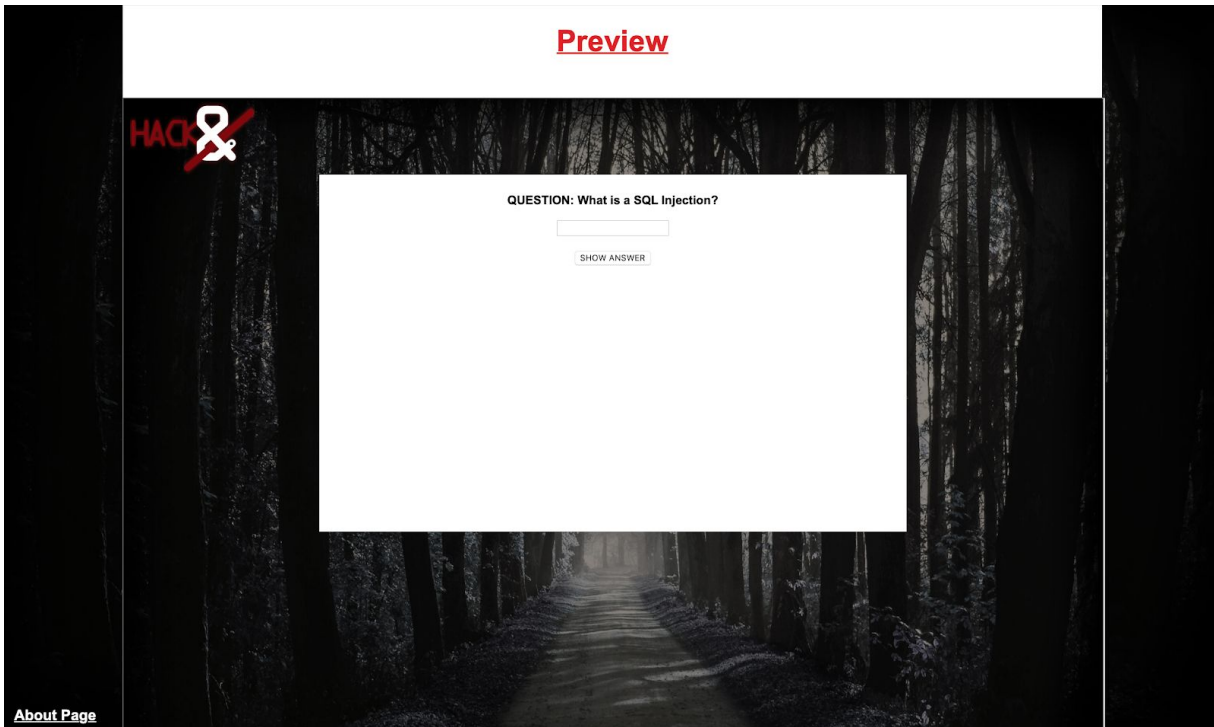
About Page

Preview

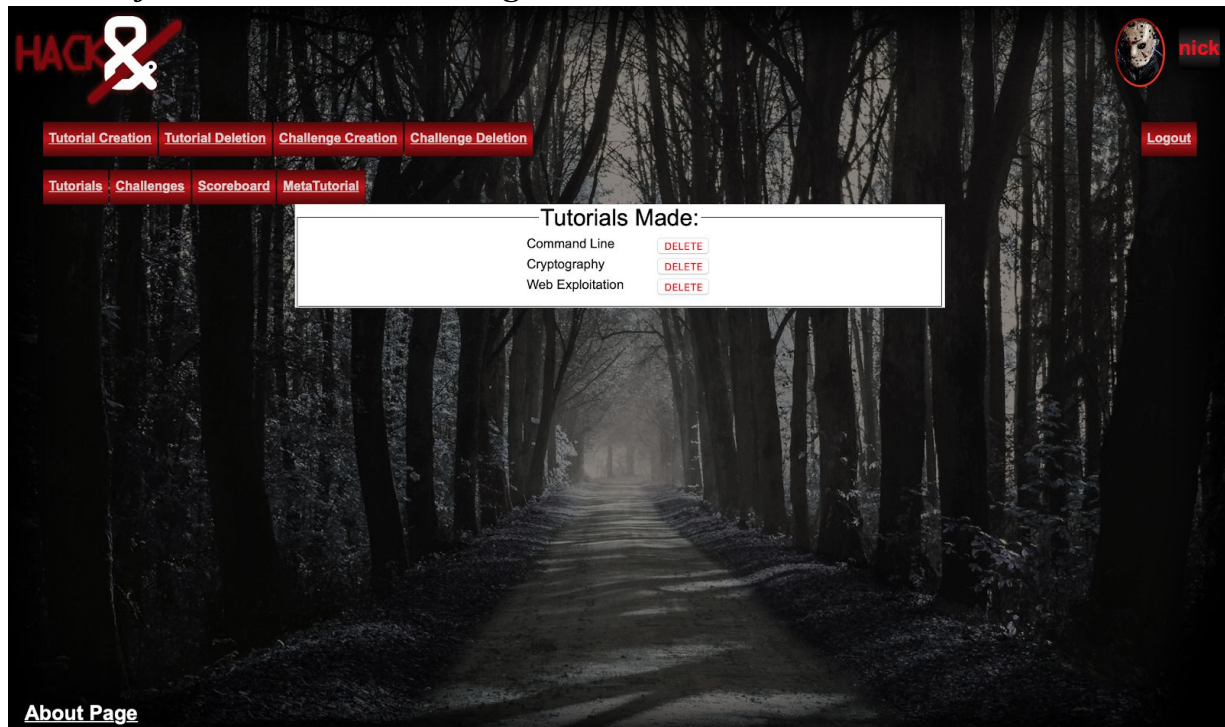
Tutorial: Web Exploitation

What is Web Exploitation? (Basic overview)

Web Exploitation is the process of obtaining sensitive data and information that is only accessible to people who are allowed to see that information. Malicious hackers are able to use website's vulnerabilities to gain access into pages and data that is only accessed by website system administrators or company executives.



j. Tutorial Deletion Page



k. Challenge Creation Page

HACK&X

Tutorial Creation Tutorial Deletion Challenge Creation Challenge Deletion

Tutorials Challenges Scoreboard MetaTutorial

Create a Challenge

Challenge Name: Web Exploitation SUBMIT

Description: Can you figure this out?

Upload your challenge files here: Choose Files page2.html

Flag: sql_injection123

Preview:

Web Exploitation
Can you figure this out?
Enter flag here:

About Page

l. Challenge Deletion Page

HACK&X

Tutorial Creation Tutorial Deletion Challenge Creation Challenge Deletion

Tutorials Challenges Scoreboard MetaTutorial

Challenges Made:

| | |
|------------------|--------|
| Command Line | DELETE |
| Cryptography | DELETE |
| Web Exploitation | DELETE |

About Page

m. Metatutorial

MetaTutorial

Creating a Tutorial

To create a tutorial, the Admin must go to the Tutorial Creation page from the Admin home page. From there, the admin must choose whether they will create a tutorial or question. If admin chooses to create a tutorial:

- Admin must upload a HTML document through the file upload button. Instructions to convert from a Microsoft Word Document to html file format are on the page.
- Each time the Admin uploads a file, it will appear in the preview section where they are then able to move sections around to their liking as well as preview it by pressing on the file box.

If the Admin wants to add a question to the tutorial:

- Admin must enter the question in the question input box
- Then they must choose if it will be a short answer or multiple choice
- If it is a short answer question, they must then add the answer
- If it is a multiple choice question, they must select how many choices there will be and what will be in each choice
- After question is completed, Admin must press the "Create Question Page" button to send it to the preview section

When Admin is done making changes, they must press the "Submit" button to finalize changes and post the tutorial.

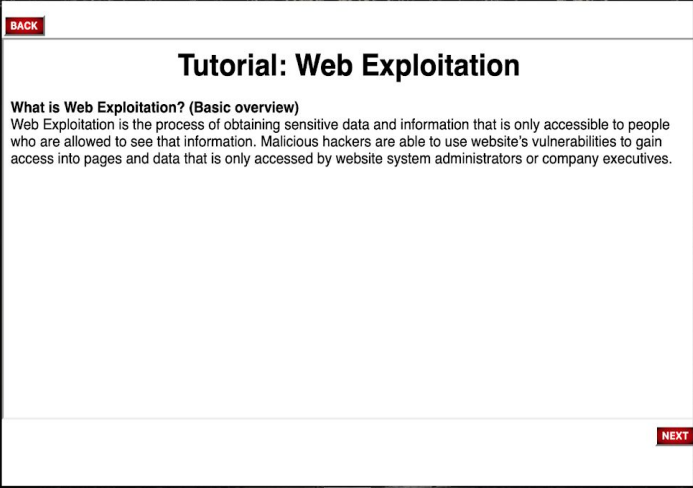

Deleting a Tutorial

To delete a tutorial, the admin must go to the Tutorial Deletion link from the Admin Home Page. From there it is a simple process. The admin must select which tutorial they want to delete by pressing the "delete" button on the right of the tutorial. A popup will then appear confirming to the admin that they are trying to delete the tutorial.

Creating a Challenge

To create a challenge, the admin must go to the Challenge Creation link from the Admin Home Page. From there the page will load with several fields to complete for the creation of

n. Tutorial Example



BACK

Tutorial: Web Exploitation

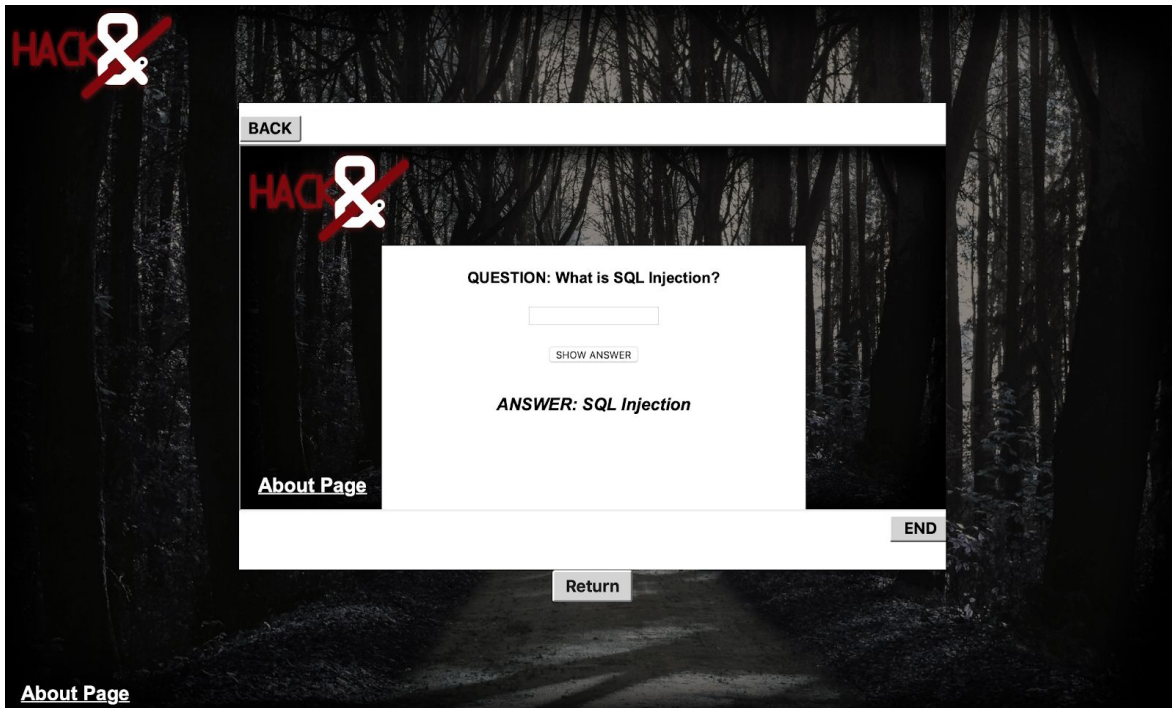
What is Web Exploitation? (Basic overview)

Web Exploitation is the process of obtaining sensitive data and information that is only accessible to people who are allowed to see that information. Malicious hackers are able to use website's vulnerabilities to gain access into pages and data that is only accessed by website system administrators or company executives.

NEXT

Return

About Page



o. Challenge Example

