Dokumentation Multi-Module Abschlussarbeit

M223 – Multi-User-Application

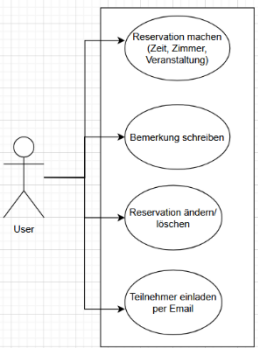
Beschreibung der Applikation

Die Anwendung die ich mit php als Backend und html & css als Frontend programmiert habe, ist ein Reservierungssystem, mit dem man in einem Formular eine Reservation erfassen kann. Nachdem man die gewünschten Uhrzeiten, Teilnehmer, Reservationsraum usw. ausgewählt hat, kann man die Reservation abgeben und man bekommt zwei einzigartige Schlüssel, ein Private-Key & ein Public-Key.

Diagramme:

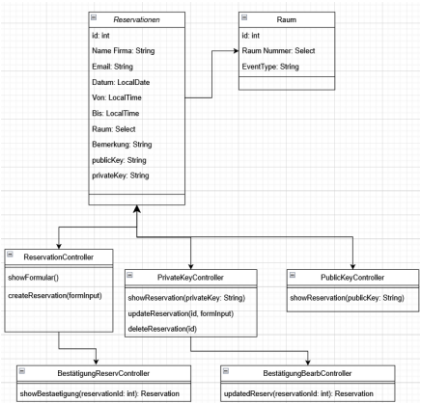
UML-Anwendungsdiagramm

Im Anwendungsdiagramm sieht man die Sachen, die der Benutzer machen kann. Der User kann in dieser Anwendung eine Reservation erstellen durch ein Formular, Bemerkungen dazu schreiben und sie bearbeiten.



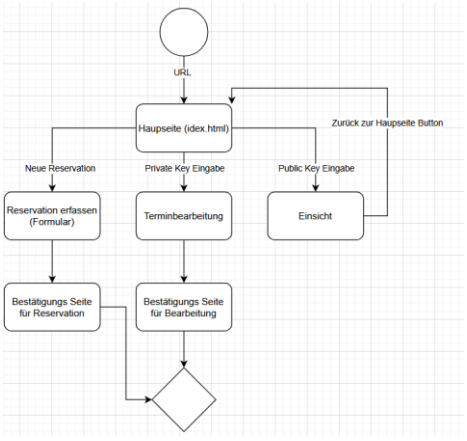
UML-Klassendiagramm

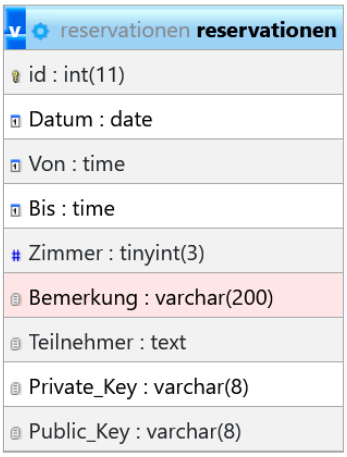
Hier stehen die Klassen, die ich für das Project geplant habe, bevor ich gestartet habe. Ich habe mich aber danach entschieden, anstatt Java php zu verwenden. Für die meisten Files habe ich keine Klassen genutzt. Für die Logik aber habe ich eine ReservationHandler Klasse erstellt, die ich nachher getestet habe.



UML-Zustandsdiagramm

Hier sehen wir den Ablauf und Reihenfolge der Anwendung. Wir starten in der Hauptseite. Wenn wir eine Reservation erstellen wollen, klicket man auf ‘Reservation erfassen’ und man geht weiter zur Formular Seite. Dort füllt man alle Felder aus und sendet es ab.





ERD

Im ERD sieht man die Struktur der MySQL Datenbank. Ich habe dieses Diagramm in phpMyAdmin unter ‘Designer’ erstellt. Hier stehen alle Columns der Tabelle ‘Reservationen’. Was speziell ist, ist dass die Spalte ‘Zimmer’ den Datentyp ‘tinyint (3) ’ hat das besagt, dass, weil im Formular ist, es ein Select Drop-Down Liste und alle Zimmer haben sowieso 3 Nummern, nur dieser wirklich nötige Speicherplatz nimmt.

Reflexion

Wie gesagt habe ich die Diagramme erstellt, bevor ich richtig mit der Anwendung begann. Ein paar Tage nachdem ich sie begonnen habe, bemerkte ich, dass vieles von was ich zu programmieren hatte bei meinem Lehrbetrieb gefragt wird, nur in java. Mir wurde seit einiger Zeit gesagt, dass ich im Praktikum hauptsächlich mit php, sql & html/css arbeiten werde. Darum habe ich mich entschieden, die Sprache zu ändern, und mit php zu arbeiten. Das ist auch der Grund, wieso das Klassendiagramm nicht mehr stimmt, weil ich es mit Java-Klassen gezeichnet habe.

Für das Projekt habe ich mehr Zeit gebraucht als ich dachte. Das ich mich entschieden habe, mit php die Anwendung zu schreiben hat mich gekostet, indem ich nach Containisierung & phpUnit Framework extra recherchieren & das noch implementieren musste. Das hat leider dazu geführt, dass ich nicht genug Zeit hatte, die Einsichtsseite & Terminbearbeitungsseite (mit Private- & Public-Key Eingabe) zu implementieren. Ich habe mich entschieden, dass ich anstatt diese zwei Seiten mithereinzubringen, mir noch bei den Aufgaben der anderen Module (kali Linux, Containisierung & Unit-Testing) Mühe zu geben. Man kann sagen ich habe also die Einsichtsseite & Terminbearbeitungsseite ‘geopfert’.

Aber trotzdem bin ich froh, dass ich alle diese zusätzliche Recherche für php gemacht habe und diese versucht habe, diese im Projekt zu implementieren, obwohl ich nicht alles geschafft habe. Das war für mich eine sehr gute Übung für das Praktikum in einen Monat und ich bin mir sicher, es wird mir zu Nutzen sein.

M347 – Containisierung

Ich habe das offizielle Image von PHP genutzt und eine selbst erstellt namens php-web. Damit das funktionierte musste ich die docker-compose.yml anpassen und ebenfalls musste ich eine kleine Änderung des Ports in der php.ini Datei machen. Was Schwierigkeiten brachte war die Verbindung zu der Datenbank von Docker. In der docker-compose.yml war mein Passwort nicht gültig. Nach langer Recherche habe ich gefunden, anstatt den Zugangsdaten kann ich «MYSQL_ALLOW_EMPTY_PASSWORD: "yes"» schreiben was funktioniert hat.

M450 – Testing

Um die Anwendung zu testen habe ich das phpUnit Framework genutzt. Am Anfang habe ich meinen Code nicht mit php-Klassen erstellt, deshalb konnte ich es auch nicht testen, denn phpUnit testet nur php-Klassen. Deswegen habe ich aus meiner submitReservation.php Datei die Logik herausgenommen und daraus eine separate php-Klasse erstellt, den ReservationHandler.php. Diese Klasse habe ich auch getestet. In ReservationHandlerTest.php habe ich 4 Unit-tests geschrieben. Die ersten zwei überprüfen die gültige Länge und Zeichen der Funktion generetekey(), die Funktion, die den zufälligen Private- & Public-Key macht, und auch eine Funktion namens isValidPostData. Ein Unit-test überprüft ob bei korrecten Angaben im Formular 'true' herauskommt (testValidPostDataTrue()) und bei dem anderen fehlt eine Eingabe und es wird geprüft, ob das gemerkt wird und 'false' herauskommt (testValidPostDataFalse()).

M183 – Sicherheit

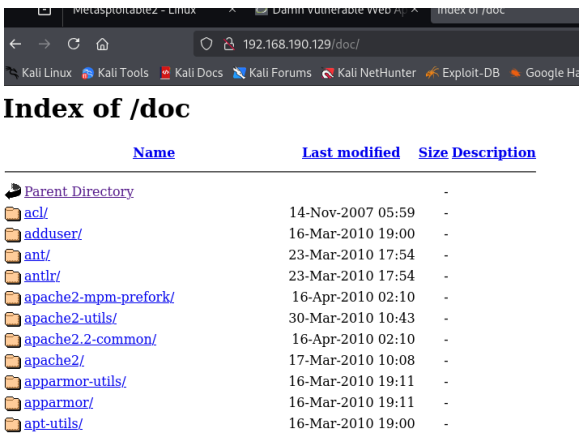
In der virtuellen Maschine kali Linux habe ich nach 5 Owasp-Gefahren gesucht. Dazu habe ich auch die VM namens Metasploitable2 heruntergeladen die ich von kali Linux angreife.

1.) A03 – SQL-Injection

Wenn man in kali Linux von der IP-Adresse von Metasploitable2 eingebt, kommt eine fast leere Seite die zu verschiedenen Anwendungen führt. Darunter ist auch ein Programm namens DVWA. In der kann man die Sicherheit auf 'low' setzten und so verschiedene Schwachstellen finden. Was ich gemacht habe, ist eine klassische SQL-Attacke zu führen, in dem ich bei der User ID « 1' OR '1'='1 » geschrieben habe. Was das macht, ist es nutzt aus das '1'='1 immer wahr ist. Und weil OR steht, wird die ganze Bedingung wahr und es werden alle Einträge zurückgegeben.

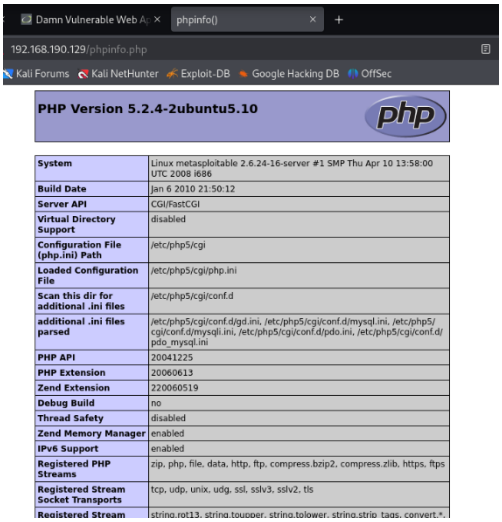
2.) A01 – Broken-Access-Control

Der zweite Fehler stammt von der ungenügenden Zugriffskontrolle. Wenn man die IP-Adresse und den gewünschten Ordner, den man sehen will, im Browser eingibt, wird der Inhalt des Ordners aufgegeben.

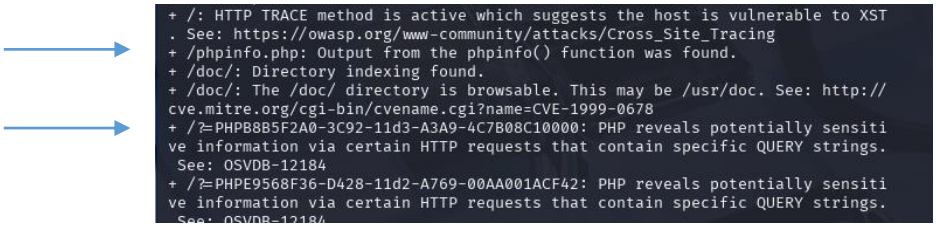


3.) A06 – Vulnerable Components – phpInfo

Ebenfalls kann man das phpinfo.php sehen, wenn man danach sucht (wieder mit IP-Adresse & Dateiname) erscheinen alle Informationen über php inklusiv Versionen, Pfade, die Struktur des Servers usw. Das sollte man nicht un-authentifiziert erreichen den für Angreifer kann das nützlich sein. Das könnte man auch zu A01 – Broken Access Control kategorisieren. Das kann man mit dem Befehl «nikto -h http://192.168.190.129» sehen, dass die Anwendung scannt.

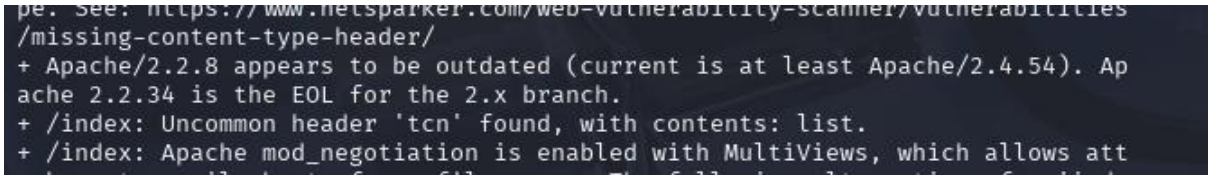


System	Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686
Build Date	Jan 6 2010 21:50:12
Server API	CGI/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php5/cgi
Loaded Configuration File	/etc/php5/cgi/php.ini
Scan this dir for additional .ini files	/etc/php5/cgi/conf.d
additional .ini files parsed	/etc/php5/cgi/conf.d/gd.ini, /etc/php5/cgi/conf.d/mysql.ini, /etc/php5/cgi/conf.d/mysqli.ini, /etc/php5/cgi/conf.d/pdo.ini, /etc/php5/cgi/conf.d/pdo_mysql.ini
PHP API	20041225
PHP Extension	20060613
Zend Extension	220060519
Debug Build	no
Thread Safety	disabled
Zend Memory Manager	enabled
IPv6 Support	enabled
Registered PHP Streams	zip, php, file, data, http, ftp, compress.bzip2, compress.zlib, https, ftps
Registered Stream Socket Transports	tcp, udp, unix, udg, ssl, sslv3, sslv2, tls
Registered Stream	string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*



4.) A06 – Vulnerable and Outdated Components – Apache

Wenn man den Befehl «nikto -h http://192.168.190.129» dann wird die Anwendung gescanned. Dort kann man sehen, dass «Apache/2.2.8 appears to be outdated». Das heisst man benutzt eine veraltete Version von Apachen und man sollte sie updaten.



5.) A07 – Identification & Authentication Failures

Wenn man auf «http://192.168.190.129:8180/» im Browser sucht, ein Port das Tomcat oft benötigt, kommt man auf der Tomcat Web Application Seite. Um bei Tomcat Änderungen zu machen, muss man auf den Tomacat Manager klicken. Da muss man den Username & Passwort eingeben. Ich konnte hereinkommen, indem ich bei beiden nur ‘tomcat’ geschrieben habe. Zuerst einmal sollten Username & Passwort nicht identisch sein. Und zudem sind diese Zugangsdaten nicht sicher. Man sollte ein eigenartiges Benutzernamen haben und ein längeres Passwort benutzten mit verschiedenen Zeichen.