Vorläufiges Thema

Bachelorarbeit
zur Erlangung des Grades *Bachelor of Science*

an der
Hochschule Niederrhein
Fachbereich Elektrotechnik und Informatik
Studiengang *Informatik*

vorgelegt von
Robert Hartings
Matrikelnummer: 1164453

Datum: 4. Juli 2020

Prüfer: Prof. Dr. Jürgen Quade
Zweitprüfer: Prof. Dr. Peter Davids

Eidesstattliche Erklärung

Name: Robert Hartings
Matrikelnr.: 1164453
Titel: Vorläufiges Thema

Ich versichere durch meine Unterschrift, dass die vorliegende Arbeit ausschließlich von mir verfasst wurde. Es wurden keine anderen als die von mir angegebenen Quellen und Hilfsmittel benutzt.

Die Arbeit besteht aus _____ Seiten.

_____ _____

Ort, Datum Robert Hartings

Inhaltsverzeichnis

1 Einleitung

Das Thema IT Sicherheit ist besonders in den letzten Jahren relevant geworden. Viele Firmen suchen Experten[it-19], welche die bestehenden und neue designten Systeme auf Sicherheitslücken prüfen und Lösungsvorschläge zur deren Behebung präsentieren. Auch werden Experten gesucht, welche die im Unternehmen bestehenden Prozesse prüfen und neue Prozesse zum Umgang mit Sicherheitslücken entwerfen.

Einen Mangel an IT-Security in privaten und öffentlichen Unternehmen beziehungsweise ein fehlendes Konzept zur Vorbeugung, Erkennung und Abwendung von Sicherheitslücken sieht man auch in jüngster Vergangenheit deutlich, nachdem beispielsweise diverse Universitäten wie Gießen, Maastricht und Bochum Ende 2019 Ziele von Hackerangriffen geworden sind. Aber nicht nur Universitäten sind betroffen, so ist neben Gerichten, Stadtverwaltungen und Krankenhäusern bereits der Deutsche Bundestag von Hackern angegriffen und kompromittiert worden.

In der Studie „Wirtschaftsschutz in der digitalen Welt“ vom 06. November 2019 des Bundesverbandes Informationswirtschaft, Telekommunikation und neue Medien e.V. Bitkom wird die aktuelle Bedrohungslage durch Spionage und Sabotage für deutsche Unternehmen untersucht. Aus dieser Studie geht hervor, dass im Jahr 2019 von Datendiebstahl, Industriespionage oder Sabotage 75% der befragten Unternehmen[1] betroffen und 13% vermutlich betroffen waren. Die Zahlen der betroffenen Unternehmen ist steigend. Im Jahre 2015 waren „nur“ 51% betroffen und 28% vermutlich betroffen. Die Unternehmen beziffern den Schaden auf 102,9 Milliarden Euro pro Jahr.[BN19]

Dass dieser Mangel auch im Lehrbetrieb angekommen ist, sieht man an neu startenden Studiengängen wie dem Bachelorstudiengang Cyber Security Management der Hochschule Niederrhein, welcher zum kommenden Wintersemester 2020/21 startet.[Hoc20]

Aber es ist zu erwähnen, dass die Hochschulen sich bereits mit dem Thema auseinandersetzen. So beschäftigt sich an der Hochschule Niederrhein das Institut für Informationssicherheit Clavis besonders mit Themen rund um das Informationssicherheitsmanagement, gestaltet aber auch Inhalte zur Vulnerabilität von (kritischer) Infrastruktur und Hacking. Das Ziel von Clavis ist die Erhöhung der Informationssicherheit von Organisationen im regionalen Umfeld der Hochschule. [Hoc] Auch hat die Hochschule Niederrhein das Thema IT-Sicherheit bereits in Ihren Lehrplan für die Studiengänge Informatik und Elektrotechnik am Fachbreich 03 Elektrotechnik und Informatik aufgenommen. So werden dort im fünften Semester in der

[1] Die Grundlage der Studie sind 1070 (2019) und 1074 (2015) befragte Unternehmen

Veranstaltung IT-Security grundlegenden Kompetenzen zum Thema IT-Sicherheit vermittelt, welche einem allgemeinen Anspruch genügen.[Hoc19]

1.1 Motivation

Neben diversen Meldungen zu erfolgreichen Angriffen auf Unternehmen und öffentliche Körperschaften und durch die Veranstaltung IT-Security im fünften Semester, besonders herauszuheben ist hier das Praktikum[2], bin ich auf das Thema IT Sicherheit aufmerksam geworden.

Die zunehmenden Vorfälle zeigen, dass ein breites Bewusstsein für IT-Sicherheit geschaffen werden muss.

Der Versuch „Catch me, if you can“ versucht dieses Bewusstsein zu schaffen, in dem die Studierenden sowohl in die Rolle des Angreifers als auch die des Schützers schlüpfen.

Das Programm, welches das Praktikum überwacht, ist bereits 10 Jahre alt und bietet meiner Meinung nach Notwendigkeiten der Modernisierung, Überarbeitung und Erweiterung. So gibt es beispielsweise heute bessere Möglichkeiten die Darstellung (Web-Oberfläche) zu realisieren.

1.2 Aufgabenstellung

Begleitend zu der Veranstaltung IT-Sicherheit für die Studiengänge Bachelor Informatik und den Bachelor Elektrotechnik des Fachbereichs 03 Elektrotechnik und Informatik der Hochschule Niederrhein werden 3 Versuche im Rahmen des Praktikums durchgeführt. Diese sollen den Studierenden praktisch Erfahrungen ermöglichen.

Der zweite Versuch „Catch me, if you can“ stellt einen Vergleichswettbewerb dar. An diesem Wettbewerb nehmen mehrere Teams teil, welche sich alle in einem gemeinsamen Netzwerk befinden. Die Aufgabe der Teams besteht darin, festgelegte Dienste abgesichert bereit zustellen, geheime Informationen sowohl auf dem eigenen Rechner als auch auf den Rechnern der anderen Teams zu finden und Schwachstellen abzusichern, umso zu verhindern, dass andere Teams an die eigenen geheimen Informationen gelangen.[Sos10, S. 2] Die geheimen Informationen sind logisch gesehen Passwörter oder private Bilder und werden durch sogenannte Flags repräsentiert. Eine Flag ist eine gehashte Zeichenfolge und hat immer die gleiche Länge.

Das Praktikum wird durch ein Auswertungs- und Überwachungssystem begleitet, welches eine objektiv nachvollziehbare Bewertung vornehmen kann und die in den Bewertungsprozess eingeflossenen Parameter dokumentiert.[Sos10, S. 2]

[2] Praktikum ist hierbei mit einer Pflichtübung vergleichbar

Ziel meiner Arbeit ist die Modernisierung und Verbesserung dieses Auswertungs- und Überwachungssystems.

In der einführenden Betrachtung (Kapitel 2) wird der aktuelle Stand des Systems, Schnittstellen zwischen Server und Client sowie der Begründung für die Veränderung dargelegt. Aus dieser einführenden Betrachtung werden dann Anforderungen abgeleitet.

Im Folgenden Kapitel 3 werden Entwürfe für die verschieden Komponenten des Servers erstellt.

Anhand der abgeleiteten Anforderungen und des Entwurfs der verschiedenen Komponenten werden im Kapitel 4 verschiedene Technologien diskutiert und passende ausgewählt.

Die Implementierung des Entwurfs mit den gewählten Technologien wird im Kapitel 5 beschrieben.

Eine kritische Auseinandersetzung mit dem Ergebnis dieser Arbeit folgt und es werden Aussichten für mögliche Veränderungen und Verbesserungen gegeben.

2 Analyse

In diesem Kapitel werden die Voraussetzungen im Labor vorgestellt, die derzeitige Implementierung des Auswertungs- und Überwachungssystems beleuchtet und kurz auf einen überwachten Client sowie dessen Schnittstellen zum System eingegangen.

2.1 Lehrveranstaltung IT-Sicherheit

Das Pflichtmodul IT-Sicherheit (ITS) ist in drei Veranstaltungen gegliedert.[Hoc19, S.30]

- Vorlesung (2 Semesterwochenstunden)
- Übung (1 Semesterwochenstunde)
- Praktikum (1 Semesterwochenstunde)

Vorlesung Die Vorlesung wird im wöchentlichen Turnus angeboten und behandelt grundlegendes Wissen zu IT-Sicherheit unter anderem in den Bereichen Gefährdung, Gegenmaßnahmen aber auch im Bereich rechtliche Gegebenheiten. Es werden Beispiele aufgezeigt, bei welchen die angesprochenen Themen gar nicht oder in einem ungenügenden Zustand umgesetzt worden sind. Die Vorlesung wird von den Veranstaltungen *Übung* (freiwillig) und *Praktikum* (verbindlich) ergänzt.

Übung Die Übungen sind freiwillig und werden im zweiwöchentlichen Turnus á 2 Stunden angeboten. Diese ermöglichen den Studierenden den durch die Vorlesung und das Selbststudium vermittelt Stoff zu vertiefen und zu festigen. Auch können dort praktische Erfahrungen gesammelt werden, von denen die Studierenden unter anderem im Praktikum profitieren können.

Praktikum Die Versuche des Praktikums finden im monatlichen Turnus (3x im Semester) á 4 Stunden statt. Bei Bestehen aller Versuche erhalten die Studierenden ihre Klausurzulassung. Jeder Versuch des Praktikums muss vorbereitet werden, dazu erhalten die Studierenden vor dem Versuch ein Hackit[1]. Nur mit erfolgreichem Absolvieren des Hackits ist es möglich am nächsten Versuch teilzunehmen.[Qua17]

[1] Aufgabe aus dem Bereich IT-Security / Hacking

2.2 Ausstattung Labor

Das Praktikum wird im Labor für Echtzeitsysteme (EZS Labor) der Hochschule Niederrhein durchgeführt.

Das Labor ist mit acht Gruppenarbeitsplätzen für Studierende sowie Arbeitsplätzen für die Betreuer und Mitarbeiter ausgestattet. Ein Arbeitsplatz der Betreuer kann zu einem neunten Gruppenarbeitsplatz umfunktioniert werden.

An einem Gruppenarbeitsplatz können 2 Studierende gleichzeitig arbeiten, da diese mit einem leistungsfähigem Desktop-PC und einem Raspberry Pi[2] sowie den dazugehörigen Peripheriegeräten (Maus, Tastatur & Monitor) ausgestattet sind. Auf den Desktop-PCs ist Ubuntu[3] und auf den Raspberry Pis ist Raspbian[4] als Betriebssystem installiert.

Auf den Desktop-PCs ist die Software VirtualBox der Firma Oracle installiert. Diese Software ermöglicht es auf dem Rechner einen weiteren Rechner zu virtualisieren. Dieser weitere PC wird Guest genannt und kann den Host, den Rechner auf dem die Software VirtualBox läuft, nicht schädigen oder beeinflussen. Sollte auf dem Guest ein Virus aktiv werden, kann dieser nicht den Host angreifen. Hierbei sollte beachtet werden, dass die Software VirtualBox Fehler haben kann oder der Nutzer Einstellungen getroffen hat, sodass der Host doch angreifbar ist.

Neben diesen Rechner steht ein Linux Server zur Verfügung, auf welchem das Auswertungs- und Überwachungssystem läuft.

Alle Rechner, auch die Guest System der Studentengruppe, sind untereinander via Ethernet verbunden.

Außerdem steht ein Beamer zur Verfügung auf dem die aktuelle Spielübersicht dargestellt werden kann.

2.3 Versuch „Catch me, if you can“

Der zweite der drei Versuche „Catch me, if you can“ wird im Rahmen eines Contest zwischen den teilnehmenden Studierenden Teams ausgetragen. Der Contest ist an ein CTF-Contest (Capture the Flag) angelehnt, nur dass die Teams Flags unter anderem auch durch das Hacken von anderen Teams erhalten können.

Der Contest ist in drei Phasen untergliedert.

1. Vorbereitung
2. Contest

[2] Einplatinencomputer mit der Größe einer Kreditkarte
[3] Ubuntu ist eine freie Linux Distribution auf Basis von Debian
[4] Abwandlung von Debian für den Raspberry Pi

3. Abschluss

Vorbereitung Die Studierenden erhalten circa 30 Minuten Zeit, um ihr System in Betrieb zu nehmen und sich mit diesem vertraut zu machen. Auch ist es möglich das System – ohne dass dieses angegriffen werden kann – abzusichern.

Contest Die Contestphase selber dauert circa 140 Minuten. In dieser Zeit dürfen die Studierenden sich untereinander Angreifen. Diese Zeit kann auch für die weitere Absicherung des eigenen Systems, die Lösung von zur Verfügung stehender Challenges sowie der Nutzung des Flagshops genutzt werden.

Abschluss Nach Ende der Contestphase müssen die Studierenden ihre Angriffe einstellen und eine Flagabgabe ist nicht mehr möglich. Die Studierenden erstellen ein Screenshot der Punkteübersicht, um diesen in ihrem Bericht aufzunehmen. Eine Nachbesprechung ist optional und ist mit maximal 30 Minuten angesetzt.

Während des Contest gelten die folgenden Regeln:

- Der Gameserver darf nicht angegriffen werden!
- Es dürfen nur die in Betrieb zu haltenden VirtualBox-Images angegriffen werden.
- Denial of Service Angriffe sind nicht erlaubt.
- Sollte eine Gruppe Root-Rechte auf einem angegriffen Rechner erlangen ist es verboten, Software auf dem Rechner zu löschen oder durch Konfiguration unbrauchbar zu machen. Sie dürfen allein die Flags auslesen.
- Flags dürfen nicht modifiziert oder gelöscht werden!
- Sämtliche Dienste müssen für den Gameserver (IP: 192.168.87.1) erreichbar bleiben!
- Das Hauptverzeichnis des HTTP-Servers /var/www/ muss für alle Rechner erreichbar bleiben, andere Verzeichnisse müssen für den internen Zugriff und extern über Username/Passwort zugänglich sein.
- SSH- und der Datenbank-Server müssen für alle erreichbar sein
- ftp-Server muss für alle erreichbar sein, Anonymous-Login ist nicht erforderlich.
- ICMP-Pakete (ping) dürfen nicht blockiert werden!
- Das Passwort des Logins »gamemaster« darf nicht zurückgesetzt werden!

[Qua17, S.9][Sos10, S.10-11]

2.4 Systemkomponenten

2.4.1 Komponenten des Servers

Im folgenden werden die verschiedenen Komponenten des Auswertungs- und Überwachungssystems in der derzeitigen Implementierung untersucht. Dabei werden Rückschlüsse auf Anforderungen gezogen sowie Schwachstellen und Verbesserungsmöglichkeiten herausgearbeitet.

Scanner

Der Scanner prüft in regelmäßigen Abständen die auf den Guest Systemen der Studierenden installierten Dienste und speichert das Ergebnis ab. Die Abstände können beim Starten des Spieles eingestellt werden. Die folgenden Dienste werden pro Team geprüft.

ScanUp Die Aufgabe dieses Scanns besteht darin zu prüfen, ob das Guest System noch für den Server erreichbar ist. Sollte das Guest System nicht erreichbar sein wird hierfür ein Strafpunkt vergeben. Aus technischer Sicht wird das Linux Kommando *ping* verwendet. An Hand des Rückgabewertes kann nachvollzogen werden, ob der Server das Guest System erreichen konnte.

ScanBubble Auf dem Guest System läuft ein selbst programmierter Bubble Server, welcher Flags via Telnet bereitstellt. Nach dem eine Flag abgeholt worden ist, erfolgt ein Timeout, sodass für eine bestimmte Zeit keine weitere Flag abgeholt werden kann. Der Bubble Server nimmt Anfragen auf dem Port *12321* für unverschlüsselte Flags und Port *12322* für verschlüsselte Flags entgegen. Der Scanner überprüft, ob eine Telnet Verbindung zu dem Port *12321* möglich ist, in dem der Scanner eine Telnet Verbindung öffnet und prüft, ob die Verbindung erfolgreich war.

ScanWebUp Jedes Guest System stellt mit Hilfe eines Apache Web Servers und php-Dateien Webseiten und Daten bereit, welche mit Hilfe von Web Clients abgerufen werden können. Dazu muss auf Port *80* der HTTP- und auf Port *443* der HTTPS Dienst laufen. Dieses verifiziert der Scanner in dem eine Socket Verbindung zu den Ports *80* und *443* geöffnet und das Ergebnis prüft wird.

ScanSQLInjectUp Dieser Scanner prüft, ob die SQL Injection des Teams erreichbar und benutzbar ist. Der Scanner sendet hierzu einen valide Kombination aus Nutzername und Passwort an den Webserver. Das Ergebnis wird dann mit dem erwarteten Ergebnis verglichen.

ScanSQLInjectSave Wie bei ScanSQLInjectUp (2.4.1) wird geprüft ob das erwartet Ergebnis zurückgeliefert wird. Besonderheit hierbei ist, dass statt einer validen Kombination aus Nutzernamen und Passwort eine SQL Injection im Nuterznamen übergeben wird. So kann geprüft werden, ob das Team die SQL Injection abgesichert hat.

ScanXSSSave Dieser Scanner prüft, ob der auf dem Guest System mögliche XSS Angriff behoben worden oder weiterhin möglich ist. Dazu wird die Webseite mit Payload, welches einen XSS Angriff darstellt, aufgerufen. In der Rückgabe wird geprüft, ob der Payload ungefilter auf der Webseite zu finden ist. Sollte diese der Fall sein, ist der XSS Angriff möglich und nicht oder unzureichend von den Studierenden abgesichert worden.

ScanSQLSave Bei diesem Scan wird kontrolliert, ob die Verbindung mit dem auf allen System voreingestellten Passwort *toor* auf dem Root Account *root* der SQL Datenbank möglich ist. Oder ob die Studierenden dieses unsichere Passwort geändert haben. Auch wird geprüft, ob das htaccess Passwort von phpMyAdmin Pafd geändert worden ist.

ScanFTPSave Auf dem Client System läuft ein FTP Server, welcher ohne Login (Nutzername & Passwort) Daten bereitstellt. Der Scanner prüft, ob ein sogenannter Anonymous Login möglich ist, in dem eine sFTP Verbindung ohne Login aufgebaut wird. Sollte die Verbindung erfolgreich sein, ist der Anonymous Login immer noch möglich.

ScanTelnetSave Ein Telnet Server wartet auf Verbindungen auf Port *23*. Da dieser Dienst nicht benötigt wird, sollen die Studierende diesen abschalten oder deinstallieren. Der Scanner prüft, ob eine Verbindung via Telnet auf Port *23* möglich ist, in dem dieser eine Verbindung via Telnet zu Port *23* aufbaut und prüft ob dieses erfolgreich war.

Generierung von Flags

Derzeitig erfolgt die Generierung der Flags sowohl auf den Clients als auch auf dem Server. Dies ist insofern notwendig, da der Server sonst nicht prüfen kann, ob die von den Studierenden abgegebenen Flags gültig sind. Für die Generierung wird folgender Algorithmus verwendet.

```
function generate($ip,$anzahl,$filename,$SALT){
        ...
        for($i=0;$i<$anzahl;$i++) {
                $seed=$SALT.$ip."Aufgabe".$i;
                $string.=md5($seed);
                $string.=";";
        }
        ...
```

}

Listing 2.1: Algorithmus zur Generiung der Flags

Dieser Algorithmus erstellt pro Team, hier durch die IP-Adresse repräsentiert, eine bestimmte Anzahl an Flags. Dazu wird ein sogenannter seed mit Hilfe der Hashfunktion MD5 gehasht. Der Seed setzt sich aus *Salt* + *IP-Adresse* + *„Aufgabe“* + *Zähler* zusammen.

Der Salt wird benötigt, um den Flags eine gewisse Lebenszeit zu geben. In der derzeitigen Implementierung enthält der Salt das aktuelle Jahr sowie das jeweilige Semester. So werden nur die Flags des aktuellen Semesters akzeptiert und eine Verwendung von Flags aus alten Semestern ist nicht möglich.

Mithilfe der IP-Adresse werden die Flags dem jeweiligen Team zugeordnet.

Der String (Zeichenfolge) „Aufgabe“ wird als Geheimnis verwendet, um das Fälschen von Flags zu verhindern.

Damit pro Team mehrere eindeutige Flags geniert werden können, wird ein sogenannter Zähler genutzt. Dieser Zähler ist auf 0 initialisiert und wird pro generierter Flag um eins erhöht, bis die benötigte Anzahl an Flags generiert worden ist.[Sos10, S.48]

Webserver

Der Webserver stellt die GUI (Graphical User Interface) für die Studierenden und Betreuer dar. Hier kann der aktuelle Punktestand angesehen werden. Auch wird in der GUI dargestellt, welches Team welchen Service abgesichert hat, inklusive der negative Punkte für nicht abgesicherte Dienste, und wie viele Strafpunkte das jeweilige Team erhalten hat.

Neben diesen Darstellungen befindet sich auf dem Server ein sogenannter Flagshop und diverse Challenges mit denen Studierende weiter Flags erhalten können.

Die Betreuer haben die Möglichkeit über die Web-GUI ein neues Spiel anzulegen, das Spiel zu starten oder zu stoppen. Auch kann von dem Spiel ein Backup erstellt werden. Neben diesen Funktionen zur Spielsteuerung können an die Teams Strafen für unfaires oder regelverletzendes Verhalte verteilt werden. Diese Strafen nehmen direkten Einfluss auf die Punkte des jeweiligen Teams. Auch besteht die Möglichkeit weiter Benutzer für das Administrationsinterface zu registrieren.

Flagshop Im Flagshop wird es den Studierenden ermöglicht weiter Flags zu kaufen. Um einen Einkauf im Flagshop durch zuführen, müssen die Teams sich vorher registrieren. Diese Registrierung erfragt eine Hand von scheinbar erforderlichen Daten. Das Format und die Erforderlichkeit der Daten kann durch eine Manipulierung der HTML Seite geändert werden. Für jeder diese Manipulation erhält der Nutzer Flags. Auch wird die Güte des angegebenen Passwortes anhand von Länge, Sonderzeichen, Groß- und Kleinbuchstaben, Ziffern bewertet und mit Flags belohnt.

Nach der Registrierung können die Studierende sich für Punkte Flags kaufen. Dazu stehen zwei Pakete mit 8 bzw. 6 Flags für den Preis von 4 Punkten pro Paket zur Verfügung. Der Preis kann auf zwei Arten reduziert werden. Entweder werden die Pakt IDs per Hand auf nicht vorhanden IDs geändert. So berechnet der Flagshop nur noch einen Preis von insgesamt 4 Punkten für beide Pakete. Um die Flags kostenlos zu erhalten, kann das *hidden input* Feld in dem der Preis zwischen gespeichert wird auf null gesetzt werden. So sind die Flags kostenlos. [Abt16, S. 63]

Auf diese Weise ist es auch möglich einen negativen Preis festzulegen und so dem eigenen Team Punkte zuzuschreiben, da eine Überprüfung in */flagshop/shop.php* nicht richtig implementiert ist. So wird nicht geprüft, ob der von dem Nutzer eingegeben Preis kleiner als null ist, sondern ob der Preis gleich null ist. Sollte diese der Fall sein, wird der Preis auf null gesetzt. Bei richtiger Implementierung würde ein negativer Preis auf null korrigiert.

```
$preis=strip_tags($_POST['preis']);
if($preis==0){
        $preis=0;
}
```

Listing 2.2: Aktuelle Prüfung des Preises

Challenges Derzeitig sind fünf Challenges implementiert, welche vom System in zufälliger Reihenfolge an interessierte Teams verteilt werden. Eine abgeschlossene oder abgebrochene Challenge, durch neu laden der Webseite oder betätigen der Zurück-Taste, kann nicht wiederholt werden. Eine Challenge kostet 10 Punkte. Nach erfolgreichem Abschließen einer Challenge gibt es 10 plus eine gewisse Anzahl an Punkten für das Absolvieren der Aufgabe. Die folgenden Challenges sind implementiert[Abt16, S.19-20].

Aufgabe 1: robots.txt Hier sollen die Studierenden anhand der robots.txt den unbekannten Ordner, welcher von Suchmaschinen nicht indexiert wird, finden und dort die geheimen Informationen auslesen.

Aufgabe 2: JavaScript-Login-Bypass Bei dieser Challenge ist die JavaScript Funktion im Quelltext versteckt. Das Verstecken ist mit einer Meldung, wie „Seitenquelltext deaktiviert“ ([Abt16]) und vielen Leerzeilen realisiert. In der aktuellsten Version von FireFox ist dieses nicht mehr möglich, da FireFox die Leerzeilen entfernt und die JavaScript Funktion oben im Quelltext zu sehen.

Aufgabe 3: Form-Modification In dieser Challenge sollen die Studierende verstehen, dass auch die Werte von Drop-Down-Menüs, Checkboxen und Radio-Buttons durch Manipulation auf nicht vorgegebene Werte geändert werden können. Deshalb ist bei diesen auch eine Serverseitige Überprüfung notwendig.

Die Aufgabe besteht darin einen bestimmten Login Namen aus einem Drop-Down-Menü auszuwählen. Da der Name nicht in dieser Liste ist, müssen die Studierenden das HTML Formular so manipulieren, dass diese den geforderten Namen auswählen können.

Aufgabe 4: JavaScript-Substrings Das Passwort, welches die Studierenden eingeben müssen, wird Client Seitig mithilfe einer JavaScript Funktion geprüft. Damit das Passwort nicht im Klartext im Quelltext steht, wird das Passwort verschleiert. So werden drei Strings Zeichen für Zeichen verglichen. Sollten die Zeichen in mindestens zwei der drei Strings gleich sein, dann gehört das Zeichen zum Passwort. Im Anschluss wird das generierte Passwort mit dem durch die Studierenden gegebenen Passwort verglichen. Sollten die Passwörter gleich sein, ist die Challenge erfolgreich abgeschlossen.

Aufgabe 5: URL-Hex-Injection Die Studierenden sollen an geheime Informationen in HEX-Wert benannten Ordner gelangen. Diese Aufgabe soll zeigen, dass Ordner die nach einen HEX-Wert benannten sind, so nicht vor Zugriffen geschützt werden können, da das HEX-Zeichen % selber durch einen HEX-Wert dargestellt werden kann.

Abgabe von Flags

Um Flags abgeben zu können, müssen die Studierenden sich mit ihren Hackits in der Web-GUI anmelden. Dort ist es möglich in einem Input Feld eine Flag synchron abzugeben. Das bedeutet, dass nach jeder Abgabe die Webseite neu geladen wird. Des Weiteren ist es nicht möglich mehrere Flags gleichzeitig abzugeben.

2.4.2 Komponenten des Clients

Da sich die Bachelorarbeit mit der Modernisierung des Auswertungs- und Überwachungssystem beschäftigt, sind nur die wichtigen Komponenten des Clients beschrieben.

Webserver des Clients

Auf den Clients läuft ein Webserver mit einigen Schwachstellen.

So ist in das Kundenbewertungsformular eine XSS Schwachstelle implementiert. Durch die Schwachstelle wird die Nutzereingabe ungefiltert in das HTML Formular übernommen. Durch diese Schwachstelle kann bösartiger Code geladen werden. Dieser Code kann dann beispielsweise Cookies, Session Tokens oder andere vertrauliche Informationen auslesen und den Angreifern übermitteln.

Eine weitere Schwachstelle stellt der sogenannte „Login zum Membersbereich“ dar. Bei einem Login Versuch wird der Benutzername und das Passwort ungefiltert in eine SQL Statement

eingefügt. So ist ein SQL Angriff auf die dahinter liegende Datenbank möglich. Durch solch einen Angriff können Daten ausgelesen werden. Diese Schwachstelle lässt sich erst beheben, wenn die Gruppe die SQL-Injection bei sich selber durchgeführt hat.

Neben diesen Schwachstellen gibt es eine Registrierung für den Flagshop. Dieses erfordert einige Eingaben, wie Name, Alter, Postleitzahl und vieles mehr. Die Eingaben sind im HTML-Formular als Pflicht markiert und haben eine Vorgabe der Form. Ein Absenden ist ohne Angabe dieser Daten nicht möglich. Die Studierenden erhalten jedoch für jede nicht getätigte und für jede nicht der Form entsprechenden Angabe Flags nach der Registrierung. Dies ist möglich, da das HTML-Formular durch die Studierenden geändert werden kann und der Server nur die Angaben bezüglich Passwort und Nutzername prüft. Diese beiden Angaben werden genutzt, um sich am Flagshop des Servers anzumelden und Flags zu erwerben. *(Siehe: 2.4.1 Flagshop)*

Außerdem stellt der Webserver eine Bildgalerie zur Verfügung in dieser befinden sich zwei Bilder, welche ebenfalls Flags enthalten.

2.5 Schnittpunkte zwischen Server und Clients

Der Server und die Clients laufen auf getrennten Systemen. Da die Studierende Schwachstellen auf ihren Clients beheben sollen, muss das Auswertungs- und Überwachungssystem auf diese Systeme zugreifen. Dadurch lassen sich die folgenden Schnittpunkte begründen.

Die Scanner prüfen vom Auswertungs- und Überwachungssystem aus, ob

- das System online ist,
- der Webserver erreichbar ist,
- der Bubble-Server erreichbar ist,
- der Login zum Membersbereich erreichbar und abgesichert ist,
- die Kundenbewertung erreichbar und abgesichert ist,
- ob das SQL Passwort geändert worden ist,
- ob der FTP Server gegen unautorisierten Zugriff abgesichert ist und
- ob der Telnet Dienst auf Port 23 abgeschaltet ist.

Des Weiteren verbinden sich die Clients beim Starten mit dem Auswertungs- und Überwachungssystem um Flags für die Flagshop-Registrierung und -Anmeldung zur Verfügung zu stellen.

2.6 Abgeleitete Anforderungen

Das Auswertungs- und Überwachungssystem muss anhand der vorhergehenden Analyse folgenden Anforderungen genügen:

- Überwachung von mindestens neun Studierendensystemen
- Ermittlung und Sicherung der Zustände von Dienste, welche auf den Studierendensystemen angeboten werden müssen
- Entgegennahme und Prüfung von Flags, inkl. der Verrechnung von (Straf-)punkten
- Ermittlung und Visualisierung der Teilergebnisse sowie des Gesamtergebnisses
- Informationsvermittlung aller Dienst- und Punkteänderungen durch unter anderem Dienststatusänderung, Flagabgabe und Strafen (fortlaufende Publikation für Studierende und Betreuer)
- Dokumentation aller Events durch Protokollierung der einzelnen Aktionen des Systems
- Bereitstellung von Challenges, damit Studierende sich weiter Punkte erarbeiten können.
- Bereitstellung eines (Flag-) Shops, bei dem mehrere Lücken genutzt werden können, um Flags zu erhalten
- Einstellungen des Spiels sollen durch Betreuer geändert werden können
- Verwaltung von Benutzern (Administratoren und Spielern)
- Zugangskontrolle für teilnehmende Studierende durch Prüfung der Hackits
- Sicherung alter Spielstände

3 Entwurf

Dieses Kapitel beinhaltet die Architektur des Systems sowie den Entwurf der einzelnen Komponenten.

Es wird die Struktur und Zusammensetzung des Systems sowie der einzelnen Komponenten skizziert. Auch werden die Anforderungen und Erwartungen an die verschiedenen Systemkomponenten dargestellt.

Bei einigen Komponenten werden auch Alternativen aufgezeigt, welche auf Grundlage der genannten Entscheidungen im Entwurf nicht verwendet worden sind.

3.1 Entwurfsziele

Bei dem Entwurf des neuen Systems sind neben den in der Analyse beschrieben Anforderungen auch folgende Ziele beachtet worden.

Beibehaltung der Features Die bereits implementierten Features Flagshop und Challenges sollen auch im neuen System verfügbar sein. Zusätzlich sollen die Studierenden aktiv angeregt werden diese auch zu nutzen.

Lose Kopplung Zwischen dem Scanner (Big Brother) und dem Webserver (Game Information System) soll eine lose Kopplung herrschen, damit die Entwicklung der beiden Komponenten unabhängig voneinander fortgesetzt werden kann.

Datenhaltung in Datenbank Die Nutzung einer Datenbank sollte aufgrund zweier Überlegungen angestrebt werden. Erstens sind alle Daten an einem Ort gebündelt. Zweitens kann die Berechnung von Punkten an die Datenbank abgeben werden. Datenbanken sind unter anderem für solche Aufgaben optimiert.

Modernisierung der GUI Das Graphical User Interface soll modernisiert werden, sodass es heutigen Standards entspricht. Auch soll hierdurch die Verständlichkeit verbessert und die Challenges sowie der Flagshop besser platziert werden.

Einheitliche Programmiersprache Eine einheitliche Programmiersprache sollte, sofern dieses Möglich ist, genutzt werden, um beispielsweise Konventionen über die gesamten Komponenten nutzen zu können. Auch vermeidet eine einheitliche Programmiersprache Fehler, welche durch verschiedene Konventionen und Syntaxen, der verschieden Programmiersprachen auftreten können.

Module/Frameworks sparsam nutzen Bei der Implementierung der Software sollte soweit dieses Notwendig und Sinnvoll ist auf bereits vorhandene Module und Frameworks zurückgegriffen werden. Durch die Minimierung von Abhängigkeiten ist die Wartung von Verwendung der Software durch Dritte leichter möglich. Auch wird die Gefahr von Bugs auslösenden Updates minimiert. Bei der Nutzung von Modulen und Frameworks sollte auf deren Verbreitung und Wartung geachtet werden, damit nicht inaktive Module/Frameworks mit eventuellen Schwächen genutzt werden.

Containerisierung Die Anwendung soll mit möglichst kleinem Wartungsaufwand überall benutzbar sein. Um dieses zu gewährleisten, sollte eine Containerisierung genutzt werden. Bei der Nutzung ist darauf zu achten, ob und mit welchen Einschränkungen diese nutzbar ist.

Ressourcen schonend Um die Ressourcen des Servers zu schonen, sollten die nicht benötigten Komponenten abgeschaltet werden. Hierbei ist der Scanner hervorzuheben, welcher nur während des Praktikums laufen muss.

3.2 Docker

Durch die Nutzung von Containerisierung ist es möglich die Anwendung agiler und skalierbarer zu betreiben. Auch werden so Abhängigkeiten zwischen den Komponenten reduziert und eine losere Kopplung erreicht, dies führt auch zu einer klareren Struktur und übersichtlicheren Programmierung. Die einzelnen Komponenten werden in eigenen Containern betrieben, so ist es möglich nicht benötigte Komponenten (temporär) abzuschalten. So ist es beispielsweise möglich eine externe Datenbank zu nutzen und die Datenbank Komponente abzuschalten. (TODO: Beispiel wirklich hier rein?)

Die Containerisierung wird mithilfe von Docker erreicht. Docker ist ein seit 2013 bestehende Open-Source Containerisierungssoftware, welche ein weite Verbreitung genießt. Sie war der de facto Standard für Containerisierung, wird in den letzten Jahren, jedoch durch Container-Orchestrierung Softwares, wie Kubernetes oder OpenShift, stückweise abgelöst.

Da eine Container-Orchestrierung viel mehr biete als im Rahmen dieses Projektes benötigt wird und eine Verwaltung und Installation von solch einer Software nicht einfach ist, wird Docker, welches den den Anforderung genügt, verwendet. So bietet Docker den Vorteil der einfacheren Bedienbarkeit und Installation sowie der benötigten Flexibilität.

Mit Hilfe von Docker Compose ist es möglich Applikationen, welche aus mehreren Containern bestehen auch Stack genannt, zu verwalten und zu betreiben. Auch ist einfacher Abhängigkeiten zwischen den Containern zu modellieren und Verbindungen zwischen Containern herzustellen.

3.3 Übersicht

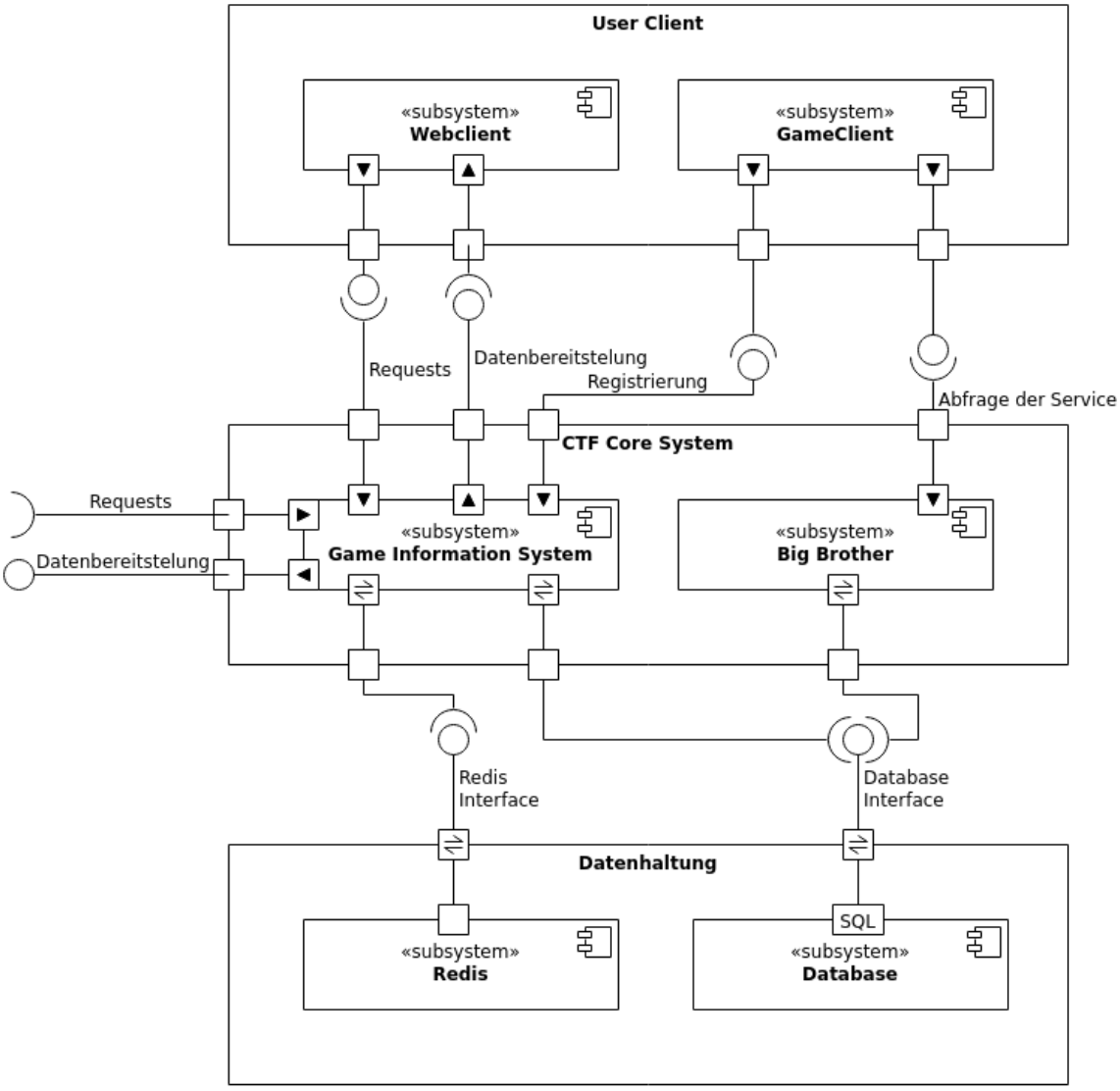

Abbildung 3.1: Übersicht über die Anwendung (Komponentendiagramm)

Die Gesamtheit des Systems (inklusive der User Clients), wie in Abbildung 3.1 zu sehen, lässt sich in drei Ebenen einteilen.

User Client Die erste Ebene *User Client* beinhaltet die auf einem User Client laufende für den Versuch relevanten Komponenten.

Bei der Komponente Webclient handelt es sich um einen geläufigen Webbrowser (häufig wird FireFox genutzt), welcher Informationen vom *CTF Core System* abruft und Daten an dieses übermittelt. Zu den Informationen gehören beispielsweise die Spielinformationen, der Spielstand aber auch teilnehmende Gruppen und Einstellungen. Der Webclient übermittelt Daten, wie gefundene Flaggen, Lösungen von Aufgaben und Änderungen an Einstellungen.

Die Komponente *GameClient* beinhaltet alle auf dem System für den Versuch installierten Anwendungen. Dazu zählen nicht nur die Dienste mit den Schwachstellen und Angriffsvektoren. Auch die clientseitige Spielverwaltungs-Software wird unter dieser Komponente zusammengefasst. Die Spielverwaltungs-Software sorgt für die richtige Konfiguration des User Clients und versteckt die Flaggen.

CTF Core System Die Ebene des *CTF Core System* besteht aus zwei Komponenten.

Zum einem aus der Komponente *Big Brother*. In der Komponente ist die Überwachung des *GameClients* mit seinen Schwachstellen und Angriffsvektoren implementiert. Die Ergebnisse dieser Überwachung werden in der *Database* festgehalten.

Zum anderen aus der Komponente *Game Information System*. Diese stellt eine Schnittstelle zwischen der Datenbank und den Nutzern dar. Mit Hilfe der Schnittstelle können Informationen unter Berücksichtigung von Berechtigungen auf einem einheitlichen Weg aus der Datenbank ausgelesen, verändert und hinzugefügt werden.

Datenhaltung Die Ebene *Datenhaltung* beinhaltet die Komponenten *Redis* und *Database*.

Die Komponente *Redis* ist nach der gleichnamigen Software Redis benannt. Bei Redis handelt es sich auch um eine Datenbank und könnte daher mit in der Komponente *Database* aufgenommen werden, die Trennung erfolgt, aber auf Grundlage der Verwendung innerhalb der Software Architektur. Die Redis-Datenbank wird als Cache verwendet und die Daten werden nicht auf der Festplatte persistiert, was in einem Performancegewinn resultiert.

Anderes als bei Redis werden in der Komponente *Database* die Daten auf eine Festplatte festgeschrieben um, diese persistent nutzen zu können.

3.4 Scanner

Im Rahmen des Versuches sollen die Studierenden ihre Systeme untersuchen und Schwachstellen und Angriffsvektoren ausfindig machen und diese dann beseitigen. Eine Überwachung wird benötigt, um zu prüfen, ob die Studierenden diese Schwachstellen und Angriffsvektoren

behoben haben. Das Abschalten beziehungsweise die Verhinderung der Verwendung eines Dienstes stellt in den meisten Fällen keine Behebung der Schwachstelle dar und wird deshalb ebenfalls geprüft. Um diese Überprüfung zu ermöglichen, wird ein Scanner benötigt, der die Dienste auf alle teilnehmenden *GameClients* abfragt und auswertet. Die bei der Überprüfung der gesammelten Daten werden benötigt, um die Servicepunkte der Gruppen zu berechnen.

3.4.1 Verteilte Scanner

Eine Idee für die Lastverteilung des Scanners ist ein verteilter Scanner. Hierbei wird ein Worker Scanner auf jedem GameClient implementiert, welcher das eigene System überwacht. Ein Master Scanner sammelt die von den Worker Scannern erstellten Ergebnisse ein und speichert diese in einer Datenbank ab.

Der Vorteile des verteilten Scanners ist die Skalierbarkeit, da der Scanner auf dem Server nur von weiteren Scannern die Ergebnisse abholen, jedoch keine Überwachung durchführen muss.

Der verteilte Scanner bringt jedoch auch einige Nachteile mit sich. So muss sichergestellt werden, dass der Scanner auf dem User Client nicht manipuliert worden ist und die Ergebnisse valide sind. Ein solche Überprüfung könnte mithilfe eines „Fingerabdrucks“ geschehen. Jedoch muss dann auch geprüft werden, ob das Programm zur Erstellung des Fingerabdrucks manipuliert worden ist. Des Weiteren muss gewährleistet werden, dass der auf dem Client laufende Scanner dieselben Antworten und Ergebnisse erhält, wie ein fremder Nutzer. Dies ist Notwendig, da die überwachten Service weiterhin von anderen Mitspielenden verwendet werden sollen.

Auf Grundlage der Nachteile, der Aufwand der Implementierung und (todo: zentraler Scanner ist ausreichend) wird ein verteilter Scanner nicht genutzt. Die Idee des zentralen Servers wird weiterverfolgt.

3.4.2 Zentraler Scanner

Die Herangehensweise des zentralen Scanners vermeidet die vorher beschriebenen Nachteile, da dem Ergebnis des Scanners vertraut werden kann und der Scanner zwangsläufig eine externe Sicht auf das System einnimmt. Dem Ergebnis kann vertraut werden, da es auf einem System ohne Einfluss der Mitspielenden berechnet wird.

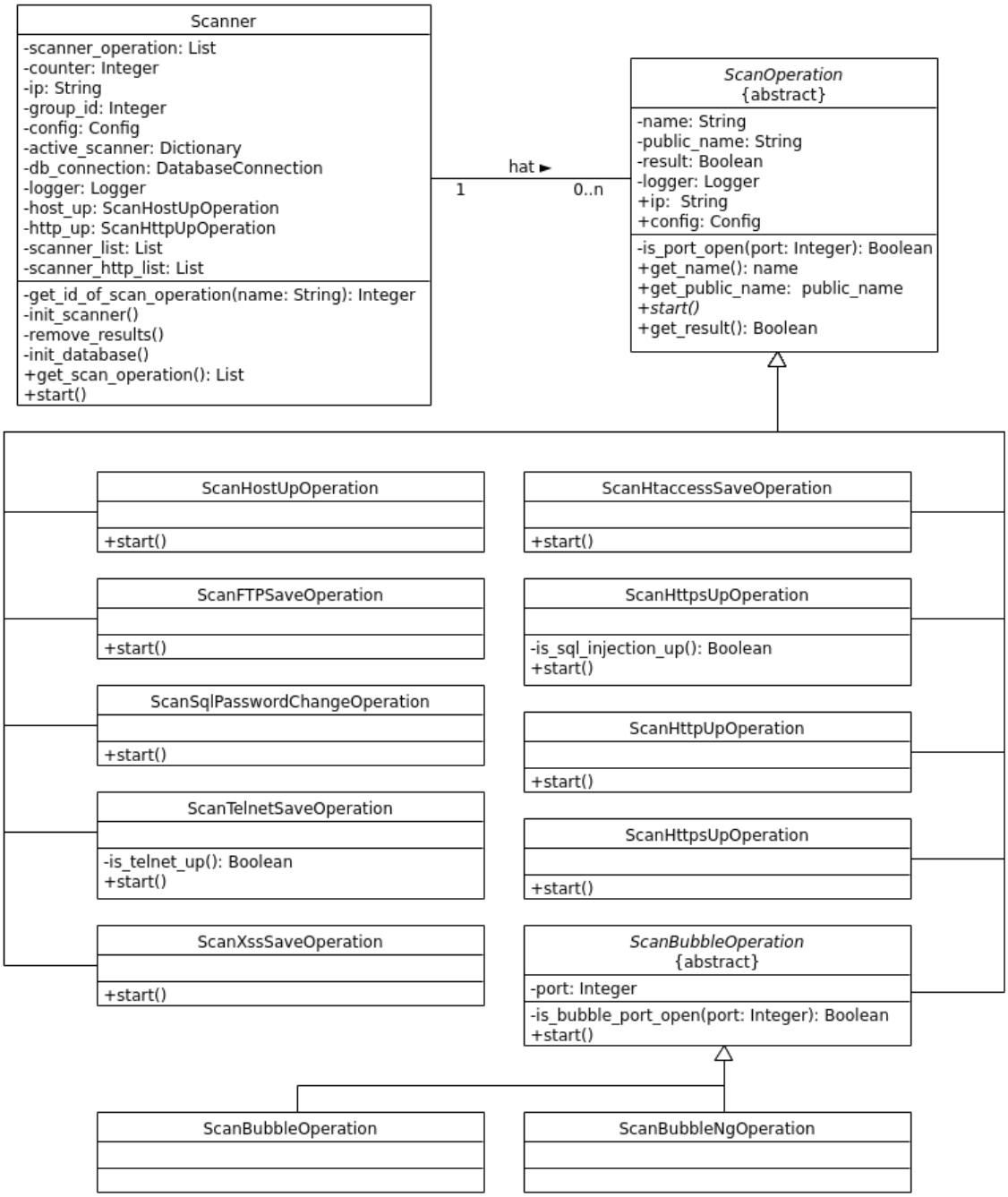

Abbildung 3.2: Klassen der Big Brother Komponente (Klassendiagramm)

Wie in Bild 3.2 erkennbar besteht die Komponente Big Brother aus der Klasse Scanner, der abstrakten Klasse ScanOperation sowie den abgeleiteten Scann-Operationen.

Die Klasse „ScanOperation“ definiert die abstrakte Funktion *start()*. Diese wird von den abgeleiteten Klassen implementiert und ermöglicht das Starten der einzelnen Scann-Operationen.

Ebenfalls speichern alle Scann-Operationen ihr Ergebnis in der privaten Variable *result*. Mithilfe der von der abstrakten Klasse implementierten Funktion *get_results()* kann der Scanner das Ergebnis der Scann-Operation auslesen. Jedes Objekt der Klasse Scanner startet 0 bis n Scann-Operationen abhängig von der Konfiguration / den aktiven Diensten. Des Weiteren stellt die abstrakte Klasse die Funktion *is_port_open* bereit, welche von den Scann-Operationen genutzt werden kann, um den Status eines Ports auf dem fremden System zu prüfen. Ein Objekt der Klasse Scanner kann pro Typ max. eine Scann-Operation starten und beinhaltet / verwaltet alle Scann-Operationen für ein Game Client.

Um mehrere Game Clients zu überwachen, werden mehrere Objekte der Klasse Scanner benötigt.

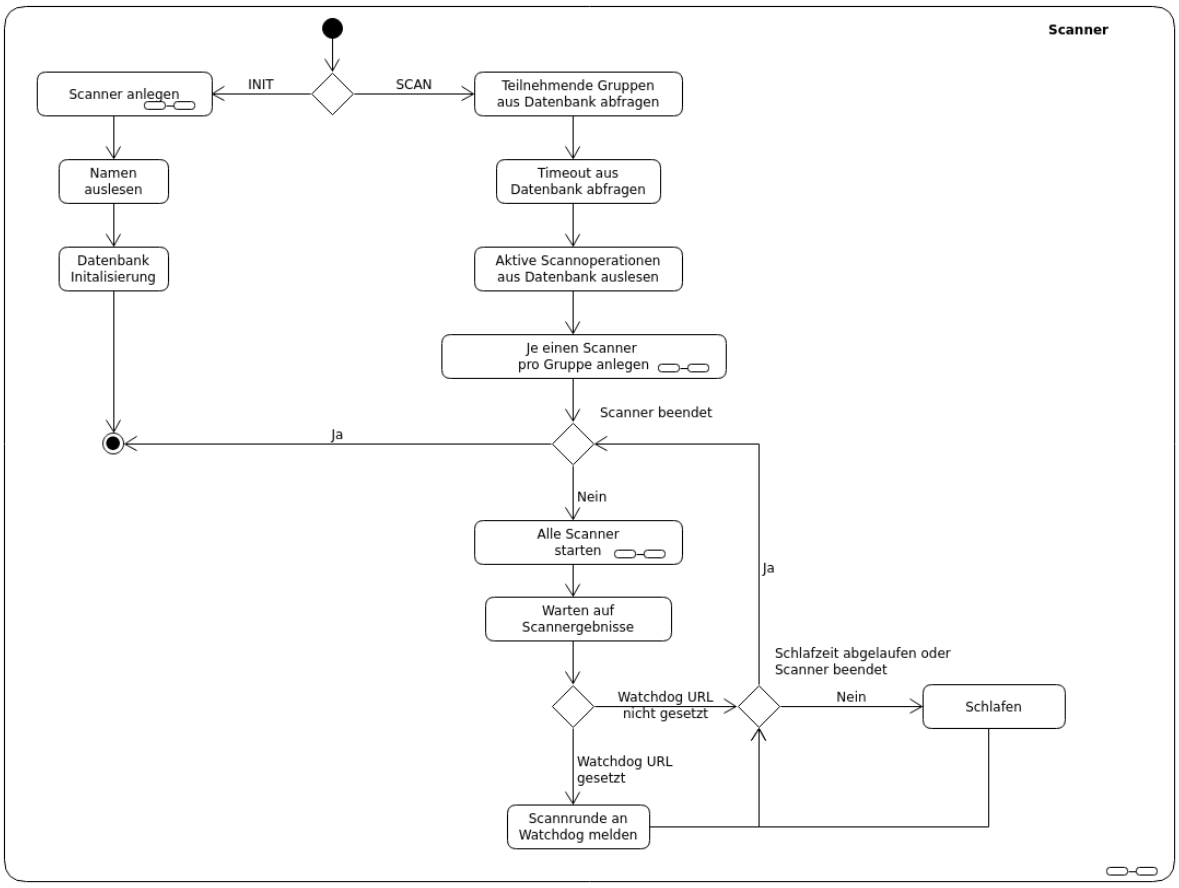

Abbildung 3.3: Ansicht des Scanners (Zustandsdiagramm)

Bei Starten des Scanners wird die zu bearbeitende Aufgabe spezifiziert.

Bekommt der Scanner die Aufgabe „INIT“, soll dieser die Service Datenbank mit den implementieren Scann-Operationen füllen, sodass Administratoren diese an- oder ausschalten können. Um die Service Datenbank zufüllen wird zunächst ein Dummy der Klasse Scanner

angelegt. Aus diesem Dummy Objekt werden von allen Scann-Operationen der Anzeigenamen und der interne Name ausgelesen. Nach dem Auslesen alle Operationen werden die erhalten Daten gebündelt in die Service Datenbank geschrieben. Danach beendet der Scanner die Verbindung zur Datenbank und endet erfolgreich mit dem Statuscode 0.

Falls die Aufgabe des Scanners „SCAN“ ist, wird der Scann der Game Clients gestartet. Hierzu werden die teilnehmenden Gruppen und der Scanner Timeout aus der Datenbank ausgelesen. Neben diesem werden die aktiven Scanner aus der Datenbank abgefragt. Sind all Informationen vorhanden, wird für jede Gruppe ein Scanner Objekt der Klasse Scanner erstellt. Bei der Erstellung werden die aktiven Scann-Operationen sowie die zu überwachende Gruppe übergeben. Das Scanner Objekt legt dann für die benötigten Scann-Operationen die jeweiligen Objekte an.

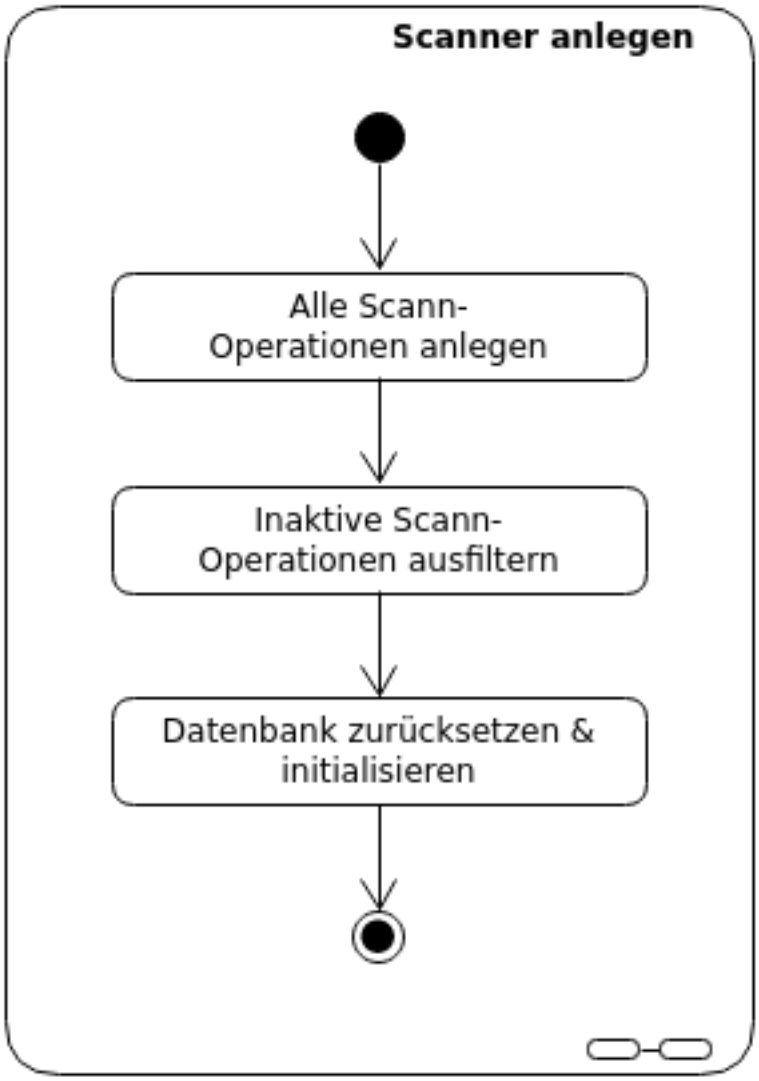

Abbildung 3.4: Erstellung eines Scanners (Zustandsdiagramm)

Nach dem Anlegen aller Scanner wird die Scann Funktion *start()* nebenläufig gestartet. Im Anschluss wird auf das Beenden der gestarteten Fuktionen gewartet. Sollten alle Funktionen abgeschlossen sein, wird geprüft, ob das Durchführen einer Scann-Runde an den REST-Server gemeldet werden soll. Ist dieses der Fall, wird der Server in Kenntnis gesetzt, dass neue Daten in der Datenbank vorhanden sind. Danach schläft der Scanner bis zum nächsten Durchlauf.

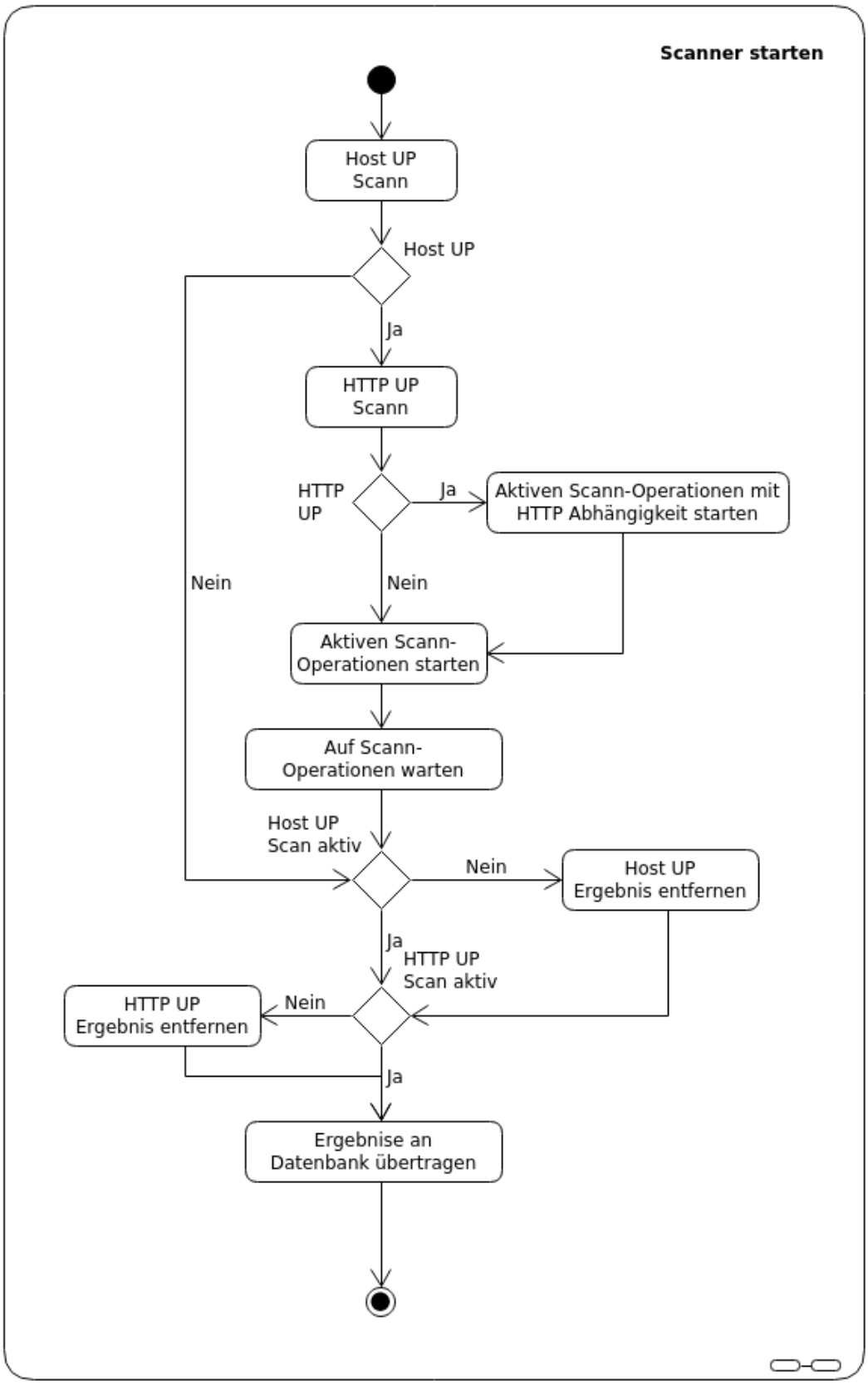
Scanner starten
Host UP Scann
Host UP
Ja
HTTP UP Scann
HTTP UP
Ja
Aktiven Scann-Operationen mit HTTP Abhängigkeit starten
Nein
Nein
Aktiven Scann-Operationen starten
Auf Scann-Operationen warten
Host UP Scan aktiv
Nein
Host UP Ergebnis entfernen
Ja
HTTP UP Scan aktiv
HTTP UP Ergebnis entfernen
Nein
Ja
Ergebnise an Datenbank übertragen

Abbildung 3.5: Starten eines Scanners (Zustandsdiagramm)

Beim Ausführen der Funktion *start()* des Scanners wird zu nächst geprüft, ob das entfernte System erreichbar ist. Sollte dieses nicht der Fall sein, werden alle nachfolgenden Scann-Operationen nicht durchgeführt, da diese fehlschlagen werden. Im Anschluss wird getestet, ob der HTTP Dienst des entfernten Systems erreichbar ist, da dieser für einige weitere Tests benötigt wird. Ist der HTTP Dienst erreichbar werden, die Scann-Operationen, welche auf dem HTTP Dienst basieren, mit in die Liste der abzuarbeiten Scann-Operationen aufgenommen. Danach werden alle verbleibenden Scann-Operationen nebenläufig gestartet. Nachdem die Scann-Operationen ihre Aufgabe abgeschlossen haben, sammelt der Scanner alle Ergebnisse ein. Falls die *Host UP Scann-Operation* oder die *HTTP UP Scann-Operation* deaktiviert ist, werden diese aus dem Ergebnis entfernt. Danach übermittelt der Scanner die gesammelten Ergebnisse zur Datenbank und beendet seine Scann-Runde.

3.4.3 Scann-Operationen

Die Scann-Operationen werden nebenläufig abgearbeitet, um so die Dauer eines kompletten Scanns zu minimieren. Eine Scann-Operation prüft genau einen Dienst / eine Schwachstelle auf dem entfernten Rechner. Die im alten System implementieren Scanns werden in die Scann-Operationen überführt. Deshalb sollen die folgenden Scann-Operationen implementiert werden.

- Host-Up
  Prüft, ob der entfernte Rechner mit Hilfe von ICMP Paketen erreichbar ist
- Bubble-Up
  Prüft, ob der Bubble Server erreichbar ist und ob die Telnet Steuerung funktioniert
- BubbleNg-Up
  Siehe Bubble-Up
- FTP-Save
  Prüft, ob der FTP Server erreichbar ist und ob die Nutzung des Anonymous Login unterbunden worden ist
- Htaccess-Save
  Prüft, ob die Kombination aus Nutzername / Passwort des Htaccess Schutz geändert worden ist
- SQL-Injection-Save
  Prüft, ob die SQL-Injection im Login zum Membersbereich verhindert worden ist
- SQL-Password-Save
  Prüft, ob das lokale Passwort des SQL-Nutzers root geändert worden ist

- Telnet-Save
  Prüft, ob der Telnet Server deaktiviert / deinstalliert worden ist
- HTTP-UP
  Prüft, ob der HTTP Dienst des entfernten Rechners nutzbar ist
- HTTPS-UP
  Prüft, ob der HTTPS Dienst des entfernten Rechners nutzbar ist
- XSS-Save
  Prüft, ob der Cross-Site-Scripting Angriff im Bewertungsformular behoben worden ist.

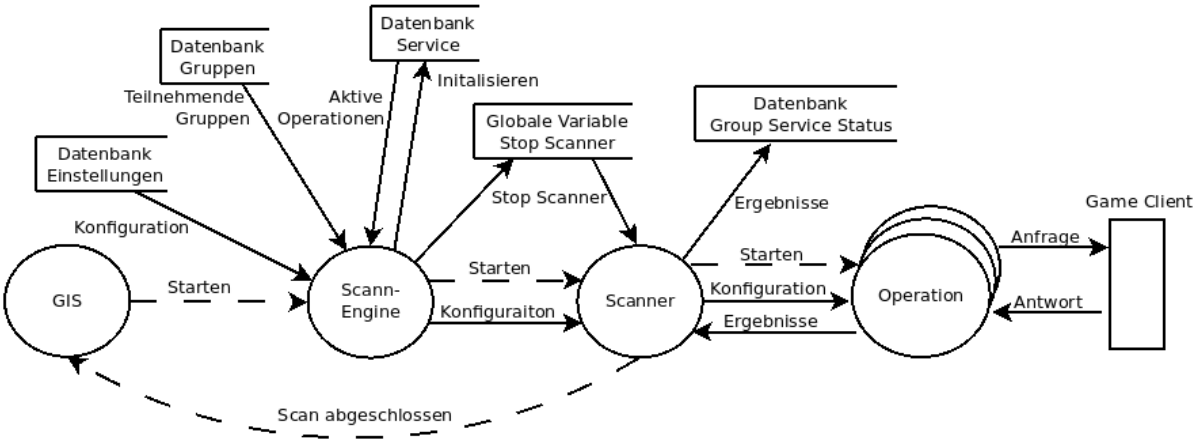

Abbildung 3.6: Datenfluss in der Scanner Komponente (Datenflussdiagramm)

Die im Datenflussdiagramm 3.6 zu sehenden, aber bisher nicht beschrieben Datenflüsse finden zwischen dem Webserver und dem Scanner oder einer Scann-Operation und dem Game Client statt. Administratoren können über den Webserver den Scanner an- und abschalten. Die Scann-Operationen frag bei dem Game Client ihren überwachten Dienst / ihre überwachte Schwachstelle an und erhalten eine Antwort zurück. Anhand dieser wird das Ergebnis der Scann-Operationen bestimmten.

3.5 Webserver

Der Webserver biete die Möglichkeit der Verwaltung des Spiels, der Abgabe von Flaggen sowie der Durchführung von Käufen im Flagshop und Challenges. Auch können Informationen zum Spiel, wie Einstellungen, Spielstand, Strafen und Teilnehmer abgerufen werden. todo

3.5.1 Verwendung mehrere Microservice

Verwendung von Microservices pro Anwendungsfall. FlagAbgabe, Nutzerverwaltun Warum habe ich das verworfen -> Zu viel Aufwand, zu wenig Value

3.5.2 Fat Webserver

Der Webserver beinhaltet sowohl die Logik als auch die Darstellung. Bei einer Anfrage an den Webserver, wird eine Antwort bestimmt und diese dann in eine Vorlage / ein HTML Dokument eingebettet. Danach wird dieses zurück an den Nutzer gesendet.

Während der Implementierung des Prototyps und des weiteren Entwurfs hat sich ein Problem mit dem Flagshop Login herausgestellt.

Für die Nutzung des Flagshops ist ein Multi-Login notwendig, da nur am Webserver eingeloggte Nutzer sich mit einem extra angelegten Account am Flagshop anmelden und diesen verwenden dürfen. Dieses ließ sich mit dem im Prototypen implementierten Session basierten Login nicht einfach umsetzen.

So ist für diesen Anwendungsfall ein Stateless Login besser geeignet, da hier die benötigten Informationen vom Client, je nach Anliegen, gesendet werden können.

Die Nutzung des Stateless Logins bietet die Perspektive der Verwendung eines Stateless Webservers sowie eines Thin Webservers. Bei einem Thin Webserver werden nur Daten und keine Repräsentation an dem Client zurückgesendet. So wird die Aufgabe der Darstellung der Daten an den Client übertragen.

Durch die Nutzung eines Thin Servers werden zwei Dinge ermöglicht.

Zum ersten ist es möglich den Client und Server unabhängig voneinander zu entwickeln, zu verändern und zu verbessern. Damit kann in Zukunft eine Iteration der Software einfacherer geschehen. Zweitens können die Studierenden eigene Clients programmieren, um mit der Anwendung zu interagieren.

3.5.3 Thin Webserver

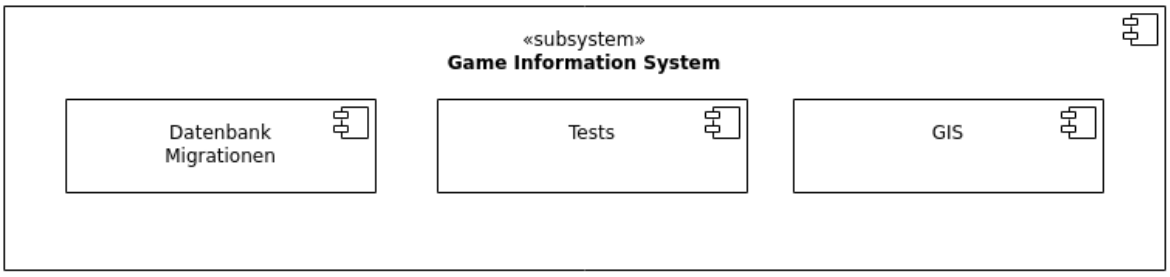

Abbildung 3.7: Rest Interface im Überblick (Komponentendiagramm)

Der Server besteht aus der Komponente *Game Information System*, welche wiederum aus drei weiteren Komponenten besteht.

Die Komponente *GIS* implementiert die gesamte Server Logik. In diesem Modul wird der eigentliche Thin Server, über das die Spieler und Betreuer mit der Anwendung interagieren können, implementiert.

In der Komponente Datenbank Migrationen sind Migrationsskripts hinterlegt, welche die Datenbank Iterationen festhalten. Diese Skripts sollen genutzt werden, um verwendete Datenbanken einfach auf den gleichen Soll-Zustand bringen zu können.

Die letzte Komponente beinhaltet Unit-Tests. Mithilfe der implementierten Unit-Tests kann die Anwendung bei späteren Änderungen auf Fehler überprüft werden kann.

Da der Thin Server nur eine Brücke zwischen Nutzer und Scanner oder Datenbank darstellt, kann von einer API (Application-Programming-Interface) gesprochen werden. Mithilfe dieser API kann beispielsweise der aktuelle Spielstand aus der Datenbank ausgelesen und Strafen eingetragen werden.

API

Um bei der Implementierung der API einem Standard / einem Vorgehen zu folgen, wird der de facto Standard für HTTP-APIs Represntational State Transfer (REST) verwendet.

Ein RESTful Interface ermöglicht, dass Ressourcen auf dem Webserver eindeutig identifizierbar sind, damit diese als Ziel von Operationen ausgewählt werden können. Des Weiteren werden einheitliche Schnittstellen genutzt. Dazu müssen Standardmethoden und -repräsentationen genutzt werden.

Durch diese Anforderungen biete das REST Interface die Möglichkeiten alle zur Verfügung stehenden Ressourcen auf die gleiche Art und Weise zu verwalten. Mit dieser Herangehensweise wird die HTTP Methode GET nicht länger zur Veränderung oder Erschaffung von Ressourcen verwendet, sondern die dafür ausgelegten HTTP-Methoden. Die für die Verwendung benötigten Daten werden auch nicht länger als Paramter der Anfrage beigefügt, sondern im Body der Anfrage übertragen.[Bei14]

Alle Routen werden mit dem Präfix */v1* versehen, um bei späteren Iterationen der API Kollisionen zu verhindern und die Nutzung der API v1 weiterhin zu ermöglichen.

Bei der Implementierung sollen die zur Verfügung stehenden HTTP Methoden benutzt werden. Das Rest-Interface soll auch mit entsprechenden HTTP Codes antworten, um die Antwort und den Erfolg einer Anfrage auch ohne Antworttext interpretieren zu können. Bei der Datenübertragung zwischen Client und Server soll das JavaScript Object Notation (JSON) Format für die Formatierung der gesendeten Daten verwendet werden.

Neben den in Tabelle 3.1 aufgezeigten Methoden gibt es noch weitere HTTP Methoden, welche keine Anwendung in dem zu implementierenden Rest-Interface erhalten.

GET	Auf Ressourcen zugreifen
POST	Neue Ressourcen erzeugen
PUT	Bestehende Ressourcen verändern
DELETE	Vorhandene Ressourcen löschen

Tabelle 3.1: Übersicht über die verwendeten HTTP Methoden

Authentifizierung

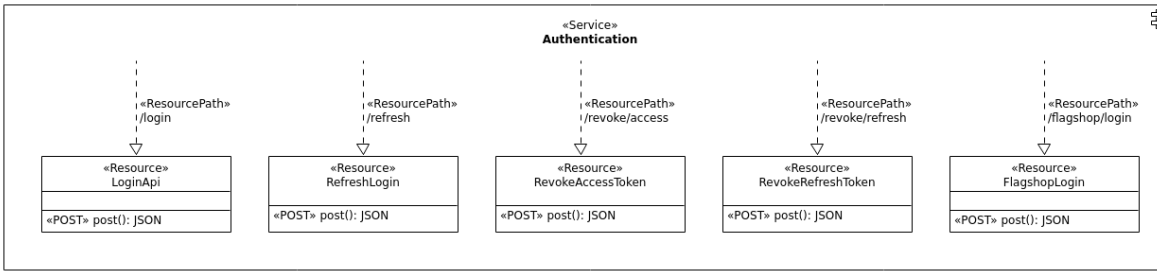

Abbildung 3.8: Übersicht über die Authentifizierung

Route	Methods
/	GET
/associate	GET, POST
/associate/<int:associate_id>	DELETE
/auth/flagshop/login	POST
/auth/login	POST
/auth/refresh	POST
/auth/revoke/access	DELETE
/auth/revoke/refresh	DELETE
/backup	GET
/backup/<int:backup_id>	GET
/challenge	GET, POST
/challenge/<int:challenge_id>	DELETE, GET, PUT
/challenge/solve	GET
/challenge/solve/<int:challenge_id>	DELETE, POST
/client	GET, POST
/client/<int:group_id>	DELETE, GET
/flag	POST
/flagshop/package	GET, POST
/flagshop/package/<int:package_id>	DELETE, PUT
/flagshop/transaction	DELETE, GET, POST
/flagshop/user	GET, POST
/flagshop/user/<user_name>	DELETE, PUT
/log	GET, POST
/log/old	GET
/match/control	DELETE, POST, PUT
/match/info	GET
/match/score	GET
/note	GET, POST
/note/<int:note_id>	DELETE, GET, PUT
/penalty	GET, POST
/penalty/<int:penalty_id>	DELETE, GET, PUT
/scanner	DELETE, GET, POST
/scanner/notify	POST
/secure	GET
/service	GET
/service/<int:service_id>	DELETE, GET, PUT
/setting	GET, PUT
/user	DELETE, GET, POST
/user/<int:user_id>	DELETE, GET, PUT
/user/import	POST

Tabelle 3.2: Übersicht über die zu implementierenden Routen

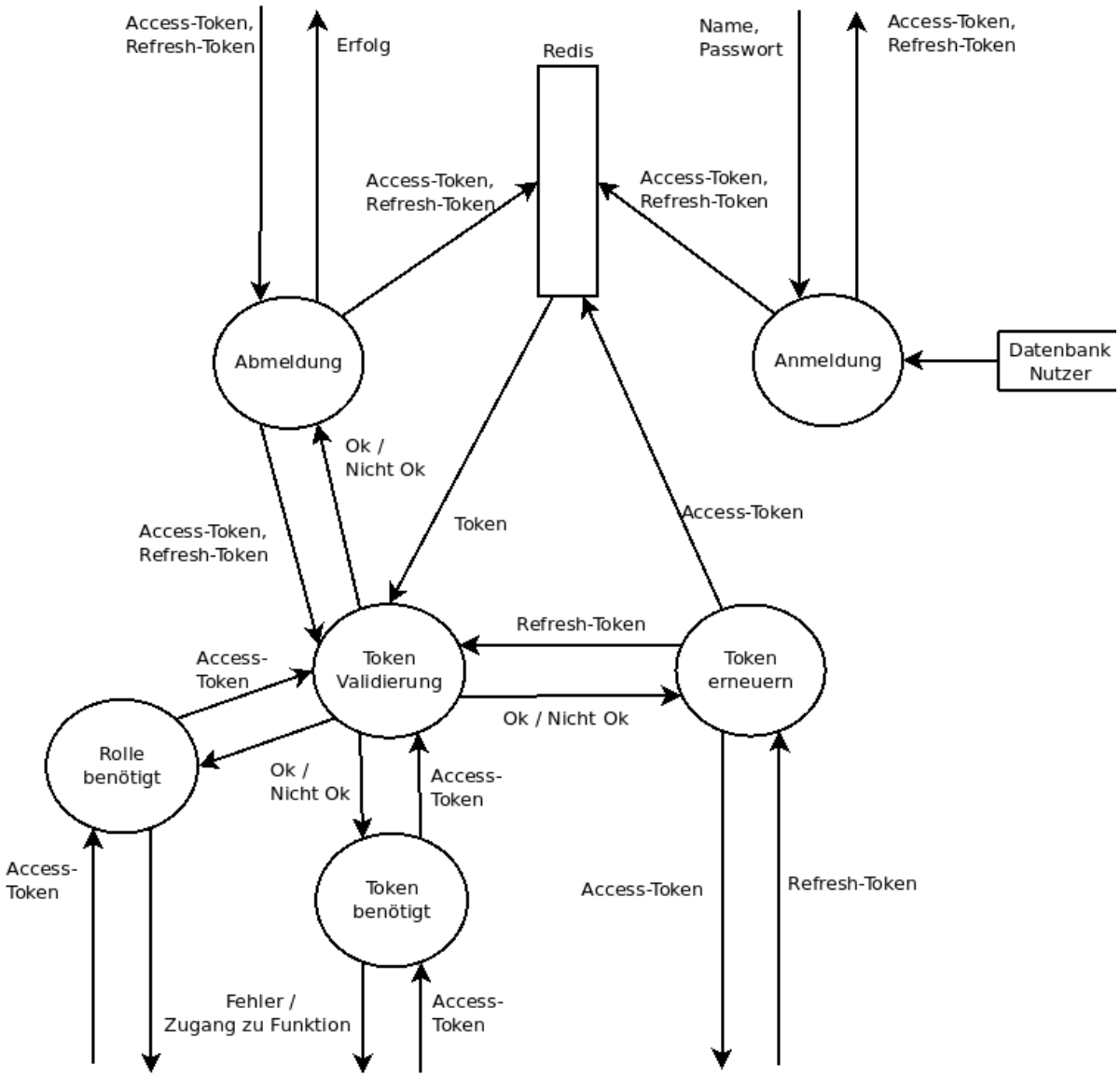

Abbildung 3.9: Datenfluss in der Security Komponente (Datenflussdiagramm)

Flaggengenierung - Änderung des Algo - Änderung des Vorgehens -> Generierung auf dem Server -> Versand an den Client (Secret nicht auf dem Client, Abfangen aller Flaggen möglich, Verschlüsselung bringt nix, da Nutzer root Rechte hat)

Challenges Challenge Informationen und Lösung der Challenge über API, Darstellung / Inhalt der Challenge über anderen Webserver (da wo auch der Webclient ausgelieftert wird), da gedanke ist nur Daten zu übertragen und keine HTML Seiten etc.

3.5.4 Reverse Proxy

- Anforderungen an den Reverse Proxy oder kommt das in die Realisierung - Warum wird ein Reverse Proxy verwendet

3.6 Datenbank

Warum relationale Datenbank Warum kein Dateisystem

3.7 Webclient

3.7.1 SPA vs MPA

Multi Page Applications Multi Page Applications, kurz MPA, ist die klassische Architektur für Webanwendungen. Bei dieser Architektur wird für jeden Request (Anfrage) an den Webserver eine neue Seite inklusive von Ressourcen wie Cascading Style Sheets (CSS)[1], JavaScript und Bildern geladen. Dieses kann mit einem Beispiel verdeutlicht werden.

Auf einer Shop-Seite befinden sich 10 Produkte inkl. Bild und Kurzbeschreibung. Wird ein Produkt ausgewählt, sendet der Client einen Request / eine Anfrage an den Webserver. Der Webserver antworte mit allen Ressourcen (siehe oben), welche für das Produkt benötigt werden. Der Client stellt dann aus den Ressourcen die Ansicht dar und das Produkt inklusive der Details ist für den Nutzer zu sehen.

Der Vorteil von MPAs ist die Optimierbarkeit für Suchmaschinen, das sogenannte SEO (Search Engine Optimization). Ein gutes SEO Rating sorgt dafür, dass die Webseite bei Suchmaschinen weit oben zu finden ist. Dies ist besonders wichtig bei Webseiten und Shops, welche um Kunden konkurrieren. Anzuführen sind hier diverse Webshops und Zeitungen.

Single Page Applications Die Single Page Applications, kurz SPA, stellt das genaue Gegenteil von MPA dar. Bei SPA besteht die Anwendung aus genau einem HTML-Dokument, dessen Inhalt bei Bedarf dynamisch nachgeladen wird. Dafür findet ein asynchroner Datenaustausch zwischen Client und Server statt, bei dem benötigte Ressourcen, wie Bilder, JavaScript und CSS ausgetauscht wird. Durch dieses Verfahren wird sicher gestellt, dass gleiche Elemente oder Ressourcen nicht erneut heruntergeladen werden müssen. Bei Änderungen werden nur Teile des DOMs[2] ersetzt und neu gerendert.

[1] Beinhalten Regeln für die Darstellung von unter anderem Webseiten
[2] Das Document Object Model repräsentiert die Webseite als Baumstruktur

Die Interaktion mit dem DOM oder auch Virtual DOM kann selber entwickelt werden. Jedoch ist hierbei zu raten, auf bereits bestehende Frameworks wie Angular (Entwickelt unter der Leitung vom Angular Team bei Google), React (Entwickelt unter der Leitung von Facebook) oder Vue (Evan You und Core Team) zurück zugreifen.

Der große Vorteil von SPA ist die Geschwindigkeit der Anwendung, da hier nur einzelne Teile ausgetauscht werden müssen. Auch bieten SPA den Vorteil, dass die Entwicklung von Front- und Backend entkoppelt wird. Das heißt, dass die Programmiere des Front- und Backends weitestgehend unabhängig von einander arbeiten können.

Die SEO Optimierung gestaltet sich schwieriger, da es sich um eine dynamische Anwendung handelt. Zur Nutzung von SPA muss im Browser JavaScript verfügbar und aktiviert sein.

Zusammenfassung Vor- und Nachteile

	SPA	MPA
Vorteile	• Sehr schnell, dank dynamischen nachladen • Entkoppelung zwischen Front- und Backend • Effizientes cachen von Daten	• MPA Architektur ist ausgereift • MPAs sind Entwickler freundlich, da ein kleiner Technologiestack benötigt wird • Ältere Browser werden unterstützt • SEO ist einfacher zu implementieren
Nachteile	• JavaScript muss im Browser verfügbar sein • Alte Browser werden nur teilweise unterstützt • Herausfordernde SEO Implementierung • Gefahr von XSS Attacken	• Anwendung sind weniger performant als MPAs • Front- und Backend haben eine starke Kopplung

Tabelle 3.3: Vor- und Nachteile SPA/MPA

Für die Entwicklung der Anwendung entscheide ich mich für die Verwendung einer SPA. Dieses geschieht unter den Gesichtspunkten der Entkopplung zwischen Front- und Backend, der Performance der Anwendung und der Zukunftssicherheit, welche meiner Meinung nach für SPA besteht. Die Nachteile vom SPA betreffen meine Anwendung gering. So ist auf den Rechnern im Labor ein moderner Webbrowser installiert und in diesem JavaScript aktiviert.

Auch handelt es sich um eine interne Anwendung, bei der die SEO Optimierung keine Rolle spielt. Einzig die Gefahr von XSS Attacken besteht, diese hoffe ich durch eine geeignete Wahl der Frontend Technologie zu reduzieren.[Mel20]

3.7.2 Mockups

3.8 Game Client

Änderungen die am Client durchgeführt werden müssen - Startgame -> register inkl. token -> Flaggen Generierung

4 Technologien

4.1 Frontend

REACT VS ANGULAR VS VUE

4.2 Backend

FLASK VS DJANGO VS EXPRESS APP

4.3 Datenhaltung

Warum PSQL als relationale Datenbank

5 Realisierung

6 Zusammenfassung & Aussicht

Anhang

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Literatur

[Abt16] Benjamin Abts. „Überarbeitung und Erweiterung eines Client- / Server-Systems zur Durchführung von ITSicherheitsschulungen (Capture the Flag)“. Bachelor Arbeit. Hochschule Niederrhein, Juni 2016. 85 S.

[Bei14] Hans Dieter Beims. *Web-Applikationen / REST*. Revision 2. 10. Dez. 2014.

[BN19] Achim Berg und Michael Niemeier. „Wirtschaftsschutz in der digitalen Welt“. In: (11. Juni 2019), S. 13.

[Hoc] Hochschule Niederrhein. *Flyer Institut Clavis*. URL: https://www.hs-niederrhein.de/fileadmin/dateien/Institute_und_Kompetenzzentren/Clavis/Flyer_Institut_Clavis__5_.pdf (besucht am 16. 05. 2020).

[Hoc19] Hochschule Niederrhein. *Modulhandbuch Vollzeit BA Informatik*. 9. Dez. 2019. URL: https://www.hs-niederrhein.de/fileadmin/dateien/FB03/Studierende/Bachelor-Studiengaenge/PO2013/modul__bi.pdf (besucht am 16. 05. 2020).

[Hoc20] Hochschule Niederrhein. *Hackern die rote Karte zeigen - Neuer Studiengang Cyber Security Management*. 7. Feb. 2020. URL: https://www.hs-niederrhein.de/startseite/news/news-detailseite/?tx_news_pi1%5Bnews%5D=18990&cHash=e849d260ecd92cf53fc9c98f6dc9edaa (besucht am 16. 05. 2020).

[it-19] it-daily.net. *IT-Security-Experten Werden Händeringend Gesucht - It-Daily.Net*. 3. März 2019. URL: https://www.it-daily.net/analysen/20773-it-security-experten-werden-haenderingend-gesucht (besucht am 16. 05. 2020).

[Mel20] Ian Melnik. *Single Page Application (SPA) vs Multi Page Application (MPA): Pros and Cons - Merehead*. 17. Apr. 2020. URL: https://merehead.com/blog/single-page-application-vs-multi-page-application/ (besucht am 04. 06. 2020).

[Qua17] Jürgen Quade. *Praktikum IT-Security*. Revision 2. 25. Sep. 2017.

[Sos10] Alexander Sosna. „Konzeption und Realisierung eines modular aufgebauten Auswertungs- und Überwachungssystems zur Durchführung von IT-Sicherheitsschulungen.“ Bachelor Arbeit. Hochschule Niederrhein, Juni 2010. 98 S.