TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

Restrukturierung der API und Serverkomponente eines mobilen Spiels zur Optimierung der Sicherheit

Bachelorarbeit

von

Daniel Theis

4. Oktober 2019

Technische Universität Kaiserslautern,
Department of Computer Science,
67653 Kaiserslautern,
Germany

Examiner: Prof. Dr. Klaus Schneider
M. Sc. Martin Köhler

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die von mir vorgelegte Arbeit mit dem Thema „Restrukturierung der API und Serverkomponente eines mobilen Spiels zur Optimierung der Sicherheit“ selbstständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Kaiserslautern, den 4.10.2019

Daniel Theis

Abstract

Trust is one of the most important things to have in a mobile game. The reason for that is, that players want to have trust into the game, that no other players are cheating. Only then they may end up spending time and possibly money on such a game.
Nowadays, many of those games are based on a server-client-system to manage game-data. If security is not taken care of enough during development, some dishonest players may take this chance to cheat.
This thesis shows the security improvements of a mobile game, which has been accomplished by developing a new server architecture. For this, we will first analyze the security-related problems of this system, for then modifying the architecture of it. For this, we use the mobile game "Game of TUK". The purpose of this game is to animate students to engage in more physical activity. To ensure security, we develop the server-component from scratch, while adapting the API to it. Doing this, we focus on server-side authentification and server-side validation of client-requests.

Zusammenfassung

Bei mobilen Spielen ist das Vertrauen in das Spiel einer der wichtigsten Faktoren, um Spieler zu binden. Ein Spieler möchte nämlich darauf vertrauen können, dass andere Mitspieler nicht betrügen. Nur so besteht die Bereitschaft, Zeit und gegebenenfalls Geld in ein solches Spiel zu investieren.
Viele dieser Spiele basieren heutzutage auf einem Server-Client-System zur Verwaltung der Spieldaten. Wird bei der Entwicklung eines solchen Spiels allerdings zu wenig Wert auf die Sicherheit dieses Systems gelegt, so eröffnet dies unehrlichen Spielern viele Möglichkeiten, dies zum Cheaten zu nutzen.
Diese Arbeit behandelt nun die Verbesserung der Sicherheit eines mobilen Spiels, die durch die Entwicklung einer neuen Architektur des Servers erreicht wird. Hierzu werden zuerst die sicherheitsbezogenen Probleme des Systems analysiert, um darauf aufbauend die Architektur dieses Systems zu modifizieren. Die Basis bildet hierbei das mobile App-Spiel „Game of TUK“, das Studenten zu mehr Bewegung animieren soll. Um die Sicherheit zu gewährleisten, wird hierfür die Serverkomponente neu entwickelt und die zugehörige API angepasst. Der Fokus liegt dabei auf der serverseitigen Authentifizierung der Spieler sowie der serverseitigen Validierung von Client-Anfragen.

Inhaltsverzeichnis

1 Einleitung

1.1 Problembeschreibung

„Bewegt studieren – Studieren bewegt“

Im Rahmen dieser Initiative der Techniker Krankenkasse und des allgemeinen deutschen Hochschulsportverbands, wurde als Reaktion auf die Studie „Gesundheit Studierender in Deutschland 2017“ [Grü+18] an der TU Kaiserslautern an einem neuen Konzept zur Bewegungsförderung getüftelt. Auslöser hierfür war, dass das Resultat dieser Studie, nach der sich drei Viertel aller Studierenden aufgrund von zu viel Stress und Leistungsdruck zu wenig bewegen, auch an der TU durch den „University Health Report 2018“ [Les+18] bestätigt werden konnte. Als Ergebnis dieser Konzeptentwicklung wurde schlussendlich das mobile App-Spiel „Game of TUK“ ins Leben gerufen.

Dieses App-Spiel wurde 2018 als Pilotprojekt am Allgemeinen Hochschulsport der TU Kaiserslautern gestartet. Das ausgelobte Ziel hiervon war es, durch Gamification die Bewegungsaktivität der Studenten zu fördern. Initiiert wurde dieses Projekt dabei von den Unisport-Mitarbeitern Julia Müller (M. Ed. Sportwissenschaft und Mathematik) und Max Sprenger (Diplom Sportlehrer und stellvertretender Leiter des Allgemeinen Hochschulsports). [18]

Ziel des Spiels ist es dabei, durch verschiedene bewegungsfördernde Aktivitäten, wie beispielsweise Fahrradfahren oder Teilnahme an Sportkursen, in der App möglichst viele Punkte zu sammeln. Als Motivationsschub wurde dabei auch eine soziale Komponente hinzugefügt. So werden die Spieler nämlich in verschiedene Gruppen – genannt „Häuser“ – aufgeteilt, für welche diese ebenfalls Punkte sammeln. Besonders erfolgreiche Häuser und sehr aktive Spieler, etwa diejenigen mit den meisten Punkten, können zudem Sachpreise gewinnen.

Dabei läuft das Spiel nicht dauerhaft, sondern beschränkt sich auf einzelne, jährlich im Sommer stattfindende Spielrunden mit einer Laufzeit von knapp vier Wochen, an deren Enden eine Preisverleihung an das Haus beziehungsweise die Spieler mit den meisten Punkten der jeweiligen Runde stattfinden.

Durch die starke soziale Komponente und die Gewinnanreize ist es dabei wichtig, garantieren zu können, dass die Ergebnisse eines Spielers respektive eines Hauses auf ehrliche Art und Weise entstanden sind. In einem Team bestehend aus Mitgliedern der Fachschaft Informatik sind wir allerdings zu dem Ergebnis gekommen, dass es in der Spielrunde von 2018 für einen Spieler vergleichsweise einfach möglich war, sich oder seinem Haus durch Cheating einen Punktevor-

teil zu verschaffen. So konnte ich gemeinsam mit den andern Informatikstudenten durch Dekompilieren der App in Erfahrung bringen, dass die Server-Client-Interaktion strukturell sehr simpel gehalten ist. Diese Struktur sorgt dann letztendlich dafür, dass Cheating ohne viele Vorkenntnisse und ohne viel Aufwand durchführbar ist.

Da diese Art des Cheatings in dieser Spielrunde tatsächlich auch vorkam und unter nicht unerheblichem Aufwand manuell korrigiert werden musste, war es elementar, diese Problematik für die darauffolgende Spielrunde zu beheben. In einem Team von drei Programmierern, von denen zwei, inklusive dem Autor dieser Arbeit, Studenten an der Universität sind, haben wir daher die Architektur und dabei insbesondere die Server-Client-Infrastruktur verbessert, sodass Cheating zwar noch möglich, aber nur mit erheblich höherem Aufwand als zuvor durchführbar ist.

1.2 Lösungsansätze

Wie später beschrieben war hierbei der Hauptbestandteil der Arbeit, einen neuen Server zu konzipieren und zu implementieren. Dieser ermöglicht eine bessere Authentifizierung und Validierung. Zur Authentifizierung wurde insbesondere die API angepasst. Für die Authentifizierung der Anfragen wird nun ein benutzerbezogenes Secret verwendet. Weiterhin wurde ein neuer Registrierungsprozesss eingeführt, welcher nun Shibboleth benutzt. Zur Validierung wurde vor allem die serverseitige Datenhaltung anders organisiert, zudem wurden außerdem Teile der Spiellogik in den Server verschoben.

1.3 Struktur der Arbeit

Ziel dieser Arbeit ist es, das Vorgehen und die Maßnahmen zur Verbesserung der Sicherheit von Game of TUK vorzustellen. Hierfür werden zuerst in Kapitel 2 der Stand der Technik sowie die ursprüngliche Architektur von 2018 dargestellt. Daraufhin erfolgt in Kapitel 3 eine Problemanalyse dieser Architektur. Darauf aufbauend wird in Kapitel 4 die neue Architektur und deren Verbesserungen gegenüber der alten aufgezeigt. Im Anschluss daran werden in Kapitel 5 beispielhaft einige Angriffsszenarien und deren Effektivität auf die neue Architektur analysiert. Schlussendlich wird in Kapitel 6 noch auf die tatsächliche Umsetzung dieser geplanten Architektur eingegangen. Dieses Kapitel schließt mit einer Übersicht der während des Projekts gemachten Erfahrungen.

2 Ausgangssituation

2.1 Was ist Game of TUK?

2.1.1 Grundgedanke

Game of TUK ist im Kern ein Spiel, das die Nutzer (speziell die Studenten der TU Kaiserslautern) zu mehr Bewegung animieren soll, indem die zugehörige App (verfügbar für iOS und Android) für verschiedene Aktivitäten auf dem Campus der Universität Punkte an die Nutzer vergibt. Dabei beschränkt sich das Spiel auf einen festgelegten Spielzeitraum von ungefähr 4 Wochen. Dies dient zum einen dazu, dass in dieser Zeit die Motivation auf einem möglichst hohen Level gehalten werden kann, zum anderen dazu, dass am Ende auch die Ergebnisse sinnvoll ausgewertet werden können.

Damit neben den reinen Punkten zusätzlich auch noch ein Wettbewerbs- beziehungsweise Kooperationsgedanke zwischen den Nutzern existiert, werden alle Mitspieler – ähnlich wie in der Bücherreihe „Harry Potter“ – in vier Häuser, welche nach Buchstaben des griechischen Alphabets benannt sind, aufgeteilt. Die Zugehörigkeit zu diesen wird dabei im generellen durch den Fachbereich des jeweiligen Studiengangs der Studenten definiert, kann jedoch unabhängig davon von jedem einzelnen Spieler frei gewählt werden.

Das Haus, welches am Ende der Spielrunde die meisten Punkte besitzt, gewinnt diese Runde und erhält gegebenenfalls einen Preis. Ebenso können Einzelspieler, welche eine der Tätigkeiten besonders gut abgeschlossen oder am Ende die meisten Gesamtpunkte haben, zusätzlich noch Einzelpreise gewinnen.

2.1.2 Funktionsweise

Die Nutzeroberfläche der App ist grundlegend in fünf Teile unterteilt:

Home Die Haupt- und Startseite der App. Hier sieht man die gesammelten Punkte der einzelnen Häuser.

Quickie Hier kann man gegen andere Mitspieler (oder einen Computergegner) bis zu 30 Mal am Tag eines von zwei kleinen Duell-Minispielen spielen um Punkte zu erhalten. Beim „Esel-Rennen“ müssen dabei zwei Tasten 50 mal abwechselnd gedrückt werden. Wer dies schneller schafft, gewinnt das Spiel. Bei „Schere, Stein, Papier“ wird das bekannte Spiel solange

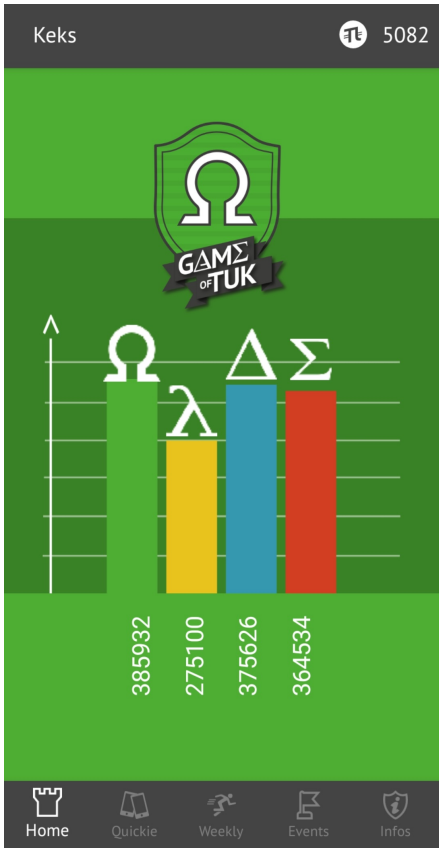

Abbildung 2.1: *Hauptbildschirm*

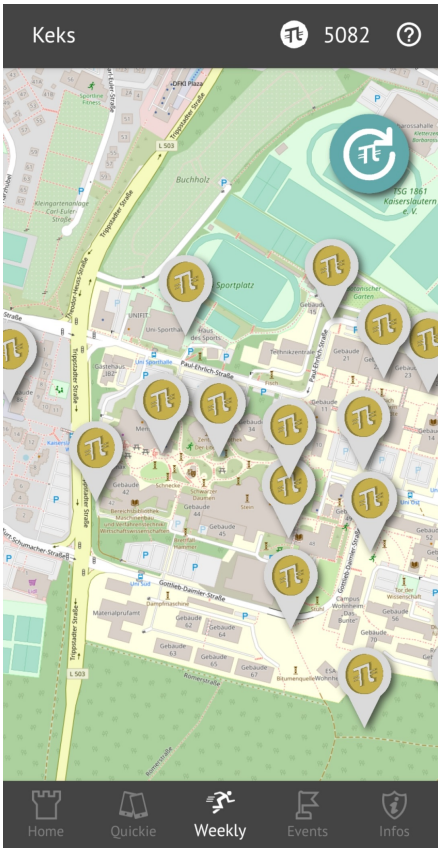

Abbildung 2.2: *Coin-Collector* [1]

gespielt bis einer von beiden Spielern 2 Runden gewonnen hat, dieser ist dann auch Gewinner dieses Minispiels.

Weekly In jeder Woche des Spielzeitraums gibt es hier eine andere Aufgabe zu meistern, die entweder einmal täglich oder über die gesamte Woche erfüllt werden kann. Diese sind:

Coin-Collector Auf einer Karte der Universität werden 20 Coins angezeigt. Der Spieler muss sich in der realen Welt an diese Orte begeben und erhält dort auf Knopfdruck Punkte. Die zugehörige Coin wird dabei von der Karte entfernt und kann an diesem Tag nicht mehr eingesammelt werden. Wenn ein Spieler es schafft alle 20 Coins zu sammeln, so bekommt er ein Bild angezeigt und erhält Bonuspunkte, wenn er den auf dem Bild gezeigten Ort aufsucht (die sogenannte Supercoin).

Fahrrad-Kilometer Der Spieler erhält für jede 100 Meter, die er auf dem Fahrrad fährt, Punkte. Dabei kann er am Tag für bis zu 20 Kilometer Distanz Punkte sammeln. Zusätzlich gibt es jeden Tag Bonuspunkte für erreichte Distanzen von 5, 10 und 20 Kilometern.

[1] Kartenausschnitt: ©OpenStreetMap-Mitwirkende

Schnitzeljagd Nach dem klassischen Prinzip der Schnitzeljagd wird dem Spieler ein Rätsel angezeigt. Besucht der Spieler den Ort auf den das Rätsel hindeutet, erhält der Spieler Punkte und den Hinweis auf den nächsten Ort. Als Unterstützung wird zudem die aktuelle Entfernung zum Zielpunkt angezeigt, sofern der Spieler sich im Umkreis von 500 Metern zum Zielpunkt befindet. Die Schnitzeljagd kann dabei insgesamt nur ein Mal absolviert werden.

Sportkurse Besucht der Spieler einen der an der Universität angebotenen Sportkurse, erhält er einen „Game of TUK Geldschein“ mit einem QR-Code. Scannt er diesen in der App ein, wird der Code invalidiert und der Spieler erhält Punkte. Da jeder eingescannte Code Punkte bringt, können somit beliebig viele Sportkurse absolviert werden.

Events Während des Spielzeitraums gibt es mehrere zeitlich und örtlich begrenzte, auf dem Campus stattfindende Events, in dem Spieler eines Hauses gegen Spieler der anderen Häuser antreten können, um Bonuspunkte für das Haus und den eigenen Spielstand zu sammeln. Die Spiele an sich sind dabei nicht im Voraus festgelegt, in der App werden daher lediglich Zeit und Ort der Events angezeigt. Die Punktevergabe erfolgt am Ende durch den Leiter des Events.

Infos Hier findet man alle wichtigen Informationen zum Spiel, unter anderem eine Spielanleitung, eine Datenschutzerklärung, eine kalendarische Übersicht des Spielzeitraums und ein News-System, über das Ankündigungen und sonstige wichtige Mitteilungen an die Spieler übermittelt werden.

Weiterhin wird zu jeder Zeit in der Kopfleiste Name und Punktestand des Spielers angezeigt. Schlussendlich können auch noch Punkte durch Registrierung, täglicher Login und Teilnahme an Umfragen zu Game of TUK verdient werden.

2.2 Stand der Technik

Besteht ein System aus Client und Server, so ist es anfällig dafür, dass ein Angreifer versucht, mit dem Server auf eine Art und Weise zu kommunizieren, die bei der Konstruktion des Clients nicht vorgesehen war. Es ist daher essentiell für die Sicherheit eines solchen Systems, dass der Server auch auf unerwartete Anfragen ein sinnvolles Verhalten zeigt.

Dabei sind die Probleme, die bei einer nicht hinreichend gegen äußere Einflüsse gesicherten Architektur auftreten können, sehr vielfältig. So könnte ein Angreifer der Zugriff auf eine verbundene Datenbank erlangt, beispielsweise Daten auslesen, diese Daten manipulieren, oder sogar die gesamte Datenbank löschen. Aber auch die reine Datenmanipulation von Anfragen kann schon sehr einfach zu Inkonsistenzen führen.

Besitzt zum Beispiel eine Webseite ein Formular mit einem Eingabefeld, in das nur echt positive Zahlen eingetragen werden dürfen, so könnte ein Angreifer beispielsweise versuchen, eine nicht legitime Anfrage mit dem Wert 0 an den Server zu senden. Wird dabei die eingegebene Zahl nur clientseitig validiert, so akzeptiert der Server die Anfrage mit den gegebenen Daten, welche zu irgendeinem Zeitpunkt zu undefiniertem Verhalten führen kann, beispielsweise wenn durch die gegebene Zahl dividiert wird. Je nachdem wie der Server mit diesem Fehler umgeht, kann das weitere Verhalten vom Anzeigen einer Fehlermeldung bis hin zum Absturz des Servers reichen.

Betrachtet man nun explizit mobile Spiele, die auf einer solchen Architektur aufbauen, so ist nach [Cho14] eines der häufigsten Ziele eines Angriffs die Manipulation von Spieldaten. Die Motivation besteht dabei oftmals darin, an Spielinhalte zu gelangen, zu denen der Spieler zu dem Zeitpunkt keinen Zugriff hat. So sind einige Inhalte, wie beispielsweise besondere Items, nur durch viele Spielstunden oder durch Einsatz von Echtgeld in sogenannten Mikrotransaktionen zu erhalten.

Dies besitzt vor allem bei Spielen mit Multiplayer-Komponente, bei der also Spieler miteinander und/oder gegeneinander spielen, eine hohe Relevanz, da in diesen der Wunsch, im Spiel zu gewinnen beziehungsweise nicht verlieren zu wollen, sehr groß ist.

Die dazu meist genutzten Techniken sind dabei unter anderem:

- Manipulation von clientseitigen Dokumenten, welche beispielsweise Spieldaten enthalten
- Verstellen der Systemzeit, um zeitbasierte Spielinhalte früher oder leichter zu erhalten
- Fälschen von Sensordaten, um darauf basierende Spielelemente zu täuschen
- Ausnutzen von API-Schwachstellen

Ein konkreteres Beispiel für einen solchen Angriff, in diesem Fall das Ausnutzen einer API-Schwachstelle, wird dabei in [Fos13] gezeigt. In dieser Präsentation wird dabei Schritt für Schritt dargestellt, wie ein Man-in-the-Middle Angriff durchgeführt wird. Bei einem Man-in-the-Middle Angriff klinkt sich dabei der Angreifer, wie der Name bereits andeutet, in die Mitte der Kommunikation zwischen Client und Server ein, und kann somit alle Nachrichten/Anfragen die zwischen diesen gesendet werden, abhören und gegebenenfalls auch manipulieren.

Das in der Präsentation genutzte Beispiel bezieht sich dabei auf das iOS Game Center, welches eine zentrale Rangliste für verschiedene Spiele-Applikationen im Apple App Store bereitstellt. Dabei kann prinzipiell jedes Spiel im App Store durch Nutzung der zugehörigen API eine solche Rangliste bereitstellen. Sofern das Spiel nur aus der App besteht, also kein dedizierter Server im Hintergrund für das Spiel genutzt wird, kann hierbei der Spieleentwickler auf

Anwendungsebene keinerlei Sicherheitsmaßnahmen ergreifen, um die Kommunikation mit dieser API zu schützen. So kann also ein Angreifer einfach eine Anfrage mit einem Punkte-Update abfangen, den Punktewert beliebig modifizieren, und die modifizierte Anfrage dann an den Server des iOS Game Center weiterleiten. Da bis einschließlich iOS 6 bei Nutzung dieser API keinerleri serverseitige Authentifizierung der Anfrage betrieben wurde, wurde somit jede modifizierte Anfrage akzeptiert. Dies wurde erst in iOS 7 durch die Nutzung eines Tokens behoben.

Wie später in Kapitel 3.4 beschrieben, ist genau dieses Problem auch so in der Game of TUK Architektur aus 2018 vorhanden. Ansonsten enthält diese auch einige der zuvor genannten Schwachstellen; hierbei sind insbesondere das Fälschen von Sensordaten, in diesem Fall des Standorts, sowie die Ausnutzung der API zu nennen. Mehr dazu folgt in Kapitel 3.

Betrachtet man nun im Gegenzug dazu einige der in [YC02] genannten Probleme, die bereits 2002 bei Online-Spielen im Allgemeinen aufgetreten sind und daher bekannt waren, so lässt sich erkennen, dass sich einige dieser Probleme respektive deren Lösungen im Laufe der Zeit stark verbessert haben. So ist beispielsweise heutzutage dem Großteil der Internetnutzer die Wichtigkeit eines sicheren Passworts bekannt, während im gleichen Zug die Entwickler die nötigen Sicherheitsmaßnahmen implementieren.

Auf der anderen Seite sind allerdings manche der Probleme auch heute noch sehr relevant. So ist beispielsweise eine Mischung der auf Seite 129 genannten Cheating-Möglichkeiten wegen „Mangel an Geheimhaltung“ und „Mangel an Authentisierung“ für das zuvor genannte Problem des iOS Game Center verantwortlich. Daran wird offensichtlich, dass auch bei heutigen Spielen, beziehungsweise Anwendungen im Allgemeinen, schon während der Entwicklung ein großer Fokus auf Sicherheit gelegt werden sollte. Dies gilt insbesondere, da die Lösungen zu diesen Problemen bereits existieren, beispielsweise die Verschlüsselung von Anfragen, und daher lediglich implementiert werden müssen.

Nutzt man beispielsweise HTTPS und SSL mit Certificate-Pinning, legt also im Quelltext der App die gültigen Zertifikate bei einer SSL-Verbindung fest, so können sogar beide Probleme auf einmal gelöst werden. Durch SSL ist die Kommunikation nämlich verschlüsselt und kann aufgrund der festgelegten Zertifikate auch nur von den Eigentümern der zertifizierten Schlüssel gelesen werden. Ein Man-in-the-Middle kennt nämlich keinen dieser Schlüssel, umgekehrt wird aber auch ein vom Angreifer erstelltes Zertifikat nicht vom Client akzeptiert.

Da sich Software und deren Design im Allgemeinen immer an die zur Verfügung stehende Hardware anpasst, eröffnen sich insbesondere mobilen Spielen auch neue Interaktionsmöglichkeiten, die beispielsweise bei einem PC-Spiel nicht umsetzbar wären. So nutzen viele mobile Spiele die neuen Möglichkeiten eines Smartphones, beispielsweise das Touch-Display durch die Nutzung von Wischgesten und Multi-Touch, oder auch die Vielzahl an Sensoren, die in einem Smartphone verbaut sind.

Aber auch hier bergen neue Möglichkeiten auch neue Schwachstellen. So setzt beispielsweise [YPK13] einige der bekannten Cheat-Möglichkeiten aus Online-Spielen, wie beispielsweise das Verstellen der Systemzeit oder das Spoofen der Identität, in den Kontext von mobilen Spielen und kommt zu dem Schluss, dass die meisten dieser Möglichkeiten in einem mobilen Kontext kritischer geworden sind. Dies bedeutet das die Erkennung einiger dieser Cheats schwieriger ist, während einige andere für einen Spieler einfacher durchzuführen sind, da er mehr Kontrolle über das Gerät besitzt.

Zusätzlich ergeben sich durch die Mobilität der Geräte auch neue Angriffsmöglichkeiten, wie das Manipulieren von Sensorwerten oder GPS-Spoofing.

Dabei sind diese durch die Offenheit der zugehörigen APIs, insbesondere beim Android-Betriebssystem, ebenfalls einfacher durchzuführen als anderweitige Angriffe. So gibt es inzwischen Apps, wie beispielsweise „Fake-GPS“, die es dem Nutzer erlauben, ohne große Mühe dem System und darauf aufbauenden Apps einen falschen Standort vorzugeben. Das genannte Paper listet aber auch direkt einige Möglichkeiten auf, um solche Manipulationen so gut wie möglich zu erkennen, beziehungsweise die Korrektheit der gegebenen Daten zu verifizieren.

So kann zum Beispiel versucht werden, durch das genutzte WLAN und/oder des Mobilfunkturms, welchen das Smartphone zu dem Zeitpunkt nutzt, den Standort des Spielers abzuschätzen. Ergeben sich nun starke Diskrepanzen zwischen dem so ermittelten Standort und dem laut System gegebenen realen Standort, so ist davon auszugehen dass der Spieler einen falschen Standort vorgibt.

Alternativ kann der Spieler auch aufgefordert werden, eine Frage zu beantworten, welche er aufgrund des Ortes an dem er vorgibt zu sein, beantworten können müsste. Kann er dies nicht, hat er vermutlich ebenfalls über seinen echten Standort hinweggetäuscht.

Wie sich in Kapitel 3 herausstellen wird, sind einige der hier genannten Fehler auch in Game of TUK 2018 vorhanden. Aus diesem Grund ist es also Ziel der Restrukturierung, diese in der Architektur enthaltenen Probleme zu beheben.

2.3 Ursprüngliche Architektur

Um über die grundlegenden Probleme der ersten App-Version reden zu können, muss zuerst einmal kurz die ursprüngliche Struktur erörtert werden. Im Allgemeinen benutzt die App für die Weeklies, also die Hauptmöglichkeit Punkte zu sammeln, die Sensoren und Dienste, im speziellen den GPS-Ortungsdienst des Handys, um Informationen über die Aktivität des Spielers zu erhalten. Je nachdem ob der Spieler für eines der Weeklies am richtigen Ort ist oder eine Bewegung des Spielers erkannt wird, erhält der Spieler nach erfolgreichem Abschluss der Aufgabe Punkte, welche zu dem Zeitpunkt erst an den Server

gesendet werden. Vorher findet in keinem der Weeklies, mit Ausnahme des Sportkurs-Weekly, eine Kommunikation mit dem Server statt.

Um hingegen die Quickies zu managen, da hierbei ja zwei Spieler durch die App miteinander interagieren müssen, ist bei jeder Aktion die den Status des Duells ändert, die Kommunikation mit dem Server erforderlich. Dabei agiert der Server hier allerdings, wie auch bei allen anderen Kommunikationsschnittstellen, lediglich als Datenbank mit einer REST-kompatiblen Zugriffs-API, durch die beide Spieler den Status des Duells abfragen können.

2.3.1 Server

Da durch die vereinbarte Festlegung des Spielzeitraums der ersten Runde die erste Version der App unter sehr großem Zeitdruck fertiggestellt werden musste, und zudem auch teilweise von unterschiedlichen Entwicklern für Android und iOS programmiert wurde, war es zweckmäßig, den Server, beziehungsweise die Schnittstelle zu diesem, so unkompliziert wie möglich zu halten. Er besteht daher effektiv aus einer Datenbank, die über ein einfaches REST-Interface erreichbar ist, und struktur-korrekte Anfragen ohne weitere Validierung des Inhalts verarbeitet. Dabei sind die einzelnen Tabellen in der Datenbank auch nicht direkt zusammenhängend, sodass bei Fehlern in der App oder gezielter Manipulation sehr einfach inkonsistente Zustände entstehen können, beispielsweise dass die Punktzahl des Hauses nicht der Punktesumme seiner Mitglieder entspricht. Dadurch ist es relativ unproblematisch, invalide Anfragen an den Server zu senden, da diese, sofern ihre Struktur korrekt ist, ohne Fehler verarbeitet werden. Die Schnittstelle ohne die Quicky-Elemente ist dabei wie folgt spezifiziert:

retrieve/	
`got_get_student_id/`	Prüft ob die übergebene Barcode-Nummer des Studenten schon in der Datenbank ist.
`got_get_nickname/`	Prüft ob der übergebene Nutzername noch nicht benutzt wird.
`got_check_money_validity/`	Prüft ob der übergebene Wert des gescannten QR-Codes noch gültig ist.
`got_get_final_team_scores/`	Gibt die Punktestände aller Häuser zurück.
`got_get_score_sum_for_nickname/`	Gibt den Punktestand des übergebenen Nutzernamens zurück.
submit/	
`got_add_student_id_info/`	Fügt die übergebene Barcode-Nummer der Datenbank hinzu.
`got_add_nickname/`	Fügt den Nutzernamen in die Datenbank ein.
`got_add_eval_info/`	Fügt, sofern angegeben, die E-Mail für die Umfrage in die Datenbank ein.
`got_write_money_info/`	Invalidiert den übergebenen QR-Code.
`got_award_user_points/`	Addiert die übergebenen Punkte zum Punktestand des übergebenen Nutzernamens.
`got_award_team_points/`	Addiert die übergebenen Punkte zum Punktestand des übergebenen Hauses.

2.3.2 Client

Parallel zum Server ist auch der Client sehr einfach gehalten. Insbesondere sind die folgenden Punkte zur Untersuchung der Sicherheit relevant:

Registrierung Der Login-Prozess der Version von 2018 funktioniert im Hintergrund komplett anders und findet bis auf die einzelnen Überprüfungen und Speicherungen fast ausschließlich auf dem Client statt. Als neuer Nutzer muss man hierbei als erstes den Barcode auf der Rückseite des Studentenausweises scannen oder alternativ die äquivalente, direkt daneben stehende Barcode-Nummer manuell eingeben. Nach einer Gegenprüfung mit dem Server, ob die Barcode-Nummer gültig ist, wird man nach dem Fachbereich für den man antreten möchte, sowie nach einem Nicknamen und optional nach einer E-Mail zur Umfragen-Teilnahme gefragt. Dabei ist die Prüfung ob ein Nutzername bereits vergeben ist, ebenso wie die Barcodevalidierung, nur clientseitig mit dem Speichern

ebendieser Daten auf dem Server verbunden. Dies bedeutet insbesondere, dass beim Speichern diese Daten nicht erneut validiert werden, was wie oben bereits beschrieben sehr einfach zu Inkonsistenzen führen kann.

Quickies Der Spieler kann auf mehrere Arten einen anderen Mitspieler zu einem Duell herausfordern:

Das QR-Duell Dies ist die schnellste Variante ein Duell zu beginnen: Ein Spieler scannt den QR-Code des anderen Spielers, welcher diesem im Menü angezeigt wird. Ist die Duellanfrage an den Server erfolgreich, so startet das Duell direkt.

Deine Feindesliste Hier können jederzeit ehemalige Duellgegner erneut herausgefordert werden.

Ärger suchen Hier kann man nach dem Nutzernamen eines beliebigen Mitspielers suchen und diesen, sofern er existiert, herausfordern.

In jedem Fall finden beide Spieler das Duell, respektive die Anfrage zum Duell ab diesem Zeitpunkt auf dem Server, der Empfänger der Anfrage muss diese jedoch, außer bei einem QR-Duell, zuerst noch annehmen. Im nun folgenden Verlauf des Duells werden die erzielten Ergebnisse zwar auch auf dem Server registriert, jedoch erfolgt die Auswertung dieser Duelle, sowie die darauf folgende Punktevergabe, wieder ausschließlich auf dem Client. Zudem sind damit die durch ein Duell erhaltenen Punkte auf dem Server nicht mit dem zugehörigen Duell gekoppelt.

Weekly Mit Ausnahme des Sportkurs-Weekly sind die einzigen Anfragen die an den Server gesendet werden, Punktevergaben an den Spieler beziehungsweise das zugehörige Haus. Die Daten zu den Weeklies, wie Standorte der Coins, Schnitzeljagd-Punkte und zugehörige Rätsel sind dabei fest in die App einprogrammiert. Ansonsten funktionieren die einzelnen Weeklies wie folgt:

Coin-Collector Auf einer Karte von Google Maps bei Android respektive Apple Maps bei iOS werden die 20 Coins dargestellt. Währenddessen läuft permanent im Hintergrund der GPS-Dienst und versorgt die App mit den aktuellen Standortdaten. Sobald ein Spieler nah genug an einer dieser Coins ist – durch Berechnung der Distanz zwischen dem aktuellen Standort des Spielers und dem Standort aller Coins – wird die Coin auf der Karte entfernt und der Spieler erhält die entsprechenden Punkte. Sobald alle Coins eingesammelt wurden, erhält der Spieler eine Mitteilung und kann an diesem Tag keine weiteren Coins einsammeln. Eine Supercoin gab es in dieser Version noch nicht.

Fahrrad-Kilometer Sobald der Spieler auf einen Startknopf drückt beginnt auch hier der GPS-Dienst das permanente Tracking, zudem wird der Initialstandort lokal gespeichert. Sobald ein vom Dienst erhaltener Standort weiter vom letzten gespeicherten Standort ent-

fernt ist als eine gegebene Minimaldistanz, wird jener Standort ebenfalls gespeichert und auf Gültigkeit überprüft. Dies bedeutet insbesondere, dass der Spieler weder zu langsam noch zu schnell gewesen sein darf, dementsprechend also mit einer Geschwindigkeit unterwegs ist, die für Fahrradfahren üblich ist. Ist der Standort gültig, so wird die gefahrene Distanz aktualisiert und vorläufig gespeichert. Dieser Prozess wiederholt sich so lange bis entweder die Maximaldistanz erreicht ist, der Spieler sich 3 Minuten außerhalb des gültigen Geschwindigkeitsbereich befindet, oder der Spieler das Tracking auf Knopfdruck in der App manuell stoppt. In jedem der Fälle wird nun die gültige, gefahrene Strecke berechnet, permanent gespeichert und schlussendlich erhält der Spieler für die gefahrene Strecke Punkte.

Schnitzeljagd Ähnlich wie bei dem Coin-Collector Weekly ist auch hier permanentes GPS-Tracking aktiv. Sobald der Spieler nah genug an dem durch das Rätsel beschriebenen Ort ist, erhält er die Punkte und das nächste Rätsel wird angezeigt. Zusätzlich erhält der Spieler aber noch eine direkte Rückmeldung wie weit er noch vom Ziel entfernt ist, sofern er sich mindestens in einem Umkreis von 500 Metern um das Ziel befindet.

Sportkurse Bei Starten des Weekly öffnet sich die Smartphone-Kamera, da die App hier nur der Erfassung der in den Sportkursen verdienten Game of TUK Geldscheine dient. Sobald die Kamera, beziehungsweise der Scanner, einen QR-Code erkennt, prüft er diesen gegen den Server auf Validität, also Existenz und Gültigkeit. Ist der Geldschein valide, so wird er in einer weiteren Server-Anfrage entwertet und in der dritten Anfrage erhält der Spieler die Punkte für das Absolvieren des Kurses.

3 Problem-Analyse

Im folgenden werden die Probleme, die sich durch die ursprüngliche Architektur ergeben haben, dargestellt; das nachfolgende Kapitel enthält dann einige Lösungsansätze zu diesen. Dabei liegt das Hauptaugenmerk auf Problemen, welche den ordnungsgemäßen Ablauf des Spiels beeinträchtigen. Dazu gehören insbesondere auch Schwachstellen, die ein Spieler der Spielrunde 2018 ausnutzen kann beziehungsweise konnte, um sich oder seinem Haus im Spiel einen Vorteil zu verschaffen.

3.1 Grundproblematik

Das Hauptproblem, aus dem die meisten anderen resultieren, ist das Fehlen einer dedizierten Serverkomponente, die alle Spieldaten validiert und in einem konsistenten Zustand hält. In der ursprünglichen Version besteht diese nämlich, wie bereits erwähnt, technisch nur aus einer Datenbank mit einem REST-Interface. REST (Representational State Transfer) beschreibt dabei ein Programmierparadigma, das oft speziell für Webservices verwendet wird. [FT00] Durch dieses ist unter anderem gewährleistet, dass alle Anfragen zustandslos sind, also ohne weitere Informationen vom Server verstanden werden können. So dürfen Anfragen beispielsweise nicht von vorherigen Anfragen abhängen. Dies wird von der Serverkomponente tatsächlich erfüllt, denn diese ist eine reine Schnittstelle, welche HTTP-Requests in Anfragen an die Datenbank umwandelt, diese ausführt, und die Ergebnisse dann wieder in HTTP-Responses verpackt an den Client schickt. Dabei ist allerdings die einzige inhaltliche Überprüfung der Requests, ob alle nötigen Parameter vorhanden sind, um mit diesen eine wohlgeformte Anfrage an die Datenbank senden zu können. So können beispielsweise auch Punkte an nicht in der Datenbank vorhandene Nutzernamen vergeben werden, zumindest insofern als dass die HTTP-Response dieselbe Antwort zurückgibt wie eine Anfrage, welche einen tatsächlich vorhandenen Nutzernamen enthält.

Ein weiteres großes Problem ist die Authentifizierung der Spieler, da zwar Benutzernamen beim Server registriert werden, dabei aber keine Spieler-Accounts im eigentlichen Sinne erstellt werden. Dies würde bedeuten, dass ein solcher Account, welcher durch den Nutzernamen eindeutig identifiziert wäre, in irgendeiner Weise an einen gewissen Spieler gebunden sein müsste, der dann auch direkt oder indirekt Zugriff auf diesen Account haben sollte, beispielsweise durch eine E-Mail. Diese fehlende Authentifizierung bedeutet insbesondere, dass bei Verlust des Handys oder Ähnlichem keine Möglichkeit besteht,

auf normalen Wege einen zuvor erstellten Spielstand fortzuführen. Umgekehrt könnte aber auch durch Manipulation der App ein fremder Spielstand übernommen, oder besser gesagt kopiert werden. Dabei würde der eigentliche, legitime Spieler dieses Spielstands zu keinem Zeitpunkt etwas davon erfahren und könnte auch im Zweifelsfall nicht nachweisen, dass er der rechtmäßige Besitzer dieses Spielstands ist und sich unter dem zugehörigen Nutzernamen in der App registriert hat. Diese Problematik wird auch in 3.4 nochmal im Detail betrachtet.

3.2 Sicherheitskritische Probleme

Wie zuvor erwähnt, werden Anfragen an den Server vor der Ausführung nur auf ihre korrekte Struktur überprüft. Damit sind durchaus auch Angriffe denkbar, die das Überlasten der Datenbank zum Ziel haben. So können zum Beispiel durch ein einfaches Skript beliebig viele Nutzernamen hinzugefügt werden. Im Gegensatz zu üblichen Datenbanken mit Nutzeraccounts müssen dabei keine gültigen E-Mails oder dergleichen zur Authentifizierung angegeben werden. Damit können innerhalb kürzester Zeit mehrere Millionen verschiedene Namen der Datenbank hinzugefügt werden. Dies führt letztlich zu einer Verlangsamung dieser, weil bei jedem Zugriff auf einen bestimmten Namen, zum Beispiel um diesem Punkte hinzuzufügen, eine Vielzahl von Namenseinträgen durchsucht werden muss, um den richtigen Eintrag in der Datenbank zu finden. Da diese Suche bei jeder einzelnen Anfrage stattfindet, verzögert sich jede Antwort, was auf Dauer zu einem Kollaps des Servers führen kann, beispielsweise wenn dieser mehr Anfragen erhält als er in der gleichen Zeit bearbeiten kann. Dies kommt effektiv einem Denial-of-Service Angriff gleich, ohne dass ein potenzieller Angreifer viel dafür tun müsste.

Außerdem ließe sich – dies ist einer der kritischsten Punkte überhaupt – das Spiel im gesamten sabotieren, indem permanent gültige, aber nicht legitime, Anfragen aller Art an den Server gesendet werden, die die Datenbank derart mit Einträgen füllen, dass legitime Anfragen nicht mehr von gefälschten unterscheidbar sind, wodurch alle Ergebnisse verfälscht wären und keine faire Bewertung der Spieler beziehungsweise der Häuser mehr möglich wäre. Dies würde zwangsläufig zu einem Abbruch des Spiels führen und das gesamte Konzept des Spiels ruinieren.

Eine weitere Problematik ist, dass diese und weitere Fehler, welche eine neue Spezifikation der API erfordern würden, nicht mehr nach Beginn der Spielrunde behoben werden können, da in der App keine Form der Versionskontrolle enthalten ist, womit es unmöglich ist, einen Spieler zwangsweise ein Update der App durchführen zu lassen. Passt man den Server jetzt an eine potenzielle neue App-Version an, so schließt man gleichzeitig die Spieler der alten Version aus, da ja nun die Kommunikationsschnittstelle eine andere ist. Lässt man nun, um dieses Problem zu beheben, den vorherigen Server parallel laufen, so können Spieler der alten Version offensichtlich immer noch dieselben

Fehler ausnutzen, ohne dass man diese, wie bereits erwähnt, zu einem Update zwingen könnte. Da es zudem kein dynamisches News-System gibt – alle Nachrichten die die Spieler erhalten können sind fest in die App einprogrammiert – gibt es zudem auch keine Möglichkeit, die Spieler überhaupt auf ein nicht geplantes Update hinzuweisen, sodass sie dies nur über Außenwege erfahren könnten, wie beispielsweise dem zum Betriebssystem gehörigen App Store.

3.3 Datenschutzkritische Probleme

Wie bereits im vorherigen Kapitel festgestellt, ist die App durch die sehr geringe serverseitige Verarbeitung von Spieldaten grundsätzlich sehr datensparsam. Das bedeutet, dass so wenige Daten wie möglich an den Server gesendet werden, insbesondere nur die Daten, die zur Abwicklung der gegebenen Anfrage auch benötigt werden. Da vor allem keinerlei Standort- oder andere Sensordaten an den Server gesendet werden, wird hierdurch das Need-To-Know-Prinzip bezüglich der Client-Server-Kommunikation umgesetzt.

Ideal wäre zudem die Gewährleistung, dass von den erhobenen Nutzerdaten sowohl server- als auch clientseitig keinerlei Weitergabe an Dritte erfolgt, damit der Datenschutz der Nutzer sichergestellt ist. In diesem Sinne stellt jedoch die Verwendung von Google Maps und Apple Maps im Coin-Collector Weekly ein Hindernis dar.

Durch die Verwendung dieser Karten können nämlich die externen Server von Google und Apple durchaus den Standort beziehungsweise den Standortverlauf nachvollziehen, da ja gerade diejenigen Kartenausschnitte geladen werden, an denen sich der Spieler befindet. Deshalb wurde auf Nachfrage der Verantwortlichen die Datenschutzerklärung der App modifiziert, sodass dort auf die Daten, die Google und Apple erhalten können, hingewiesen wurde. Wünschenswert wäre aber eine Lösung, die keine oder zumindest so wenige Daten wie möglich an Dritte versendet.

3.4 Spielfairness

Das wichtigste Ziel von Game of TUK ist es, Nutzer zu mehr Bewegung zu motivieren. Nur solange Spieler der App, respektive der Spielstruktur als Ganzes vertrauen, werden sie sich von ihr motivieren lassen. Daher ist die Spielfairness eines der wichtigsten Elemente bei dieser Art des kompetitiven Spiels. Sobald ein Spieler nämlich das Gefühl hat, dass irgendein Gegenspieler nicht fair spielt und dies nicht geahndet wird, so verliert er sehr schnell die Motivation an dem Spiel. Da bei Game of TUK ebenfalls Spieler beziehungsweise verschiedene Parteien – die Häuser – gegeneinander spielen, ist es daher essentiell, eine solche Spielfairness sicherzustellen.

In dieser Version der App wurden allerdings einige Schwachstellen entdeckt, die teilweise auch von nicht technikaffinen Spielern sehr einfach ausgenutzt werden können, um im Spiel zu schummeln oder sich und seinem Haus anderweitig einen Vorteil zu verschaffen. Dabei ist zumeist die schwache serverseitige Validierung ein großes Problem.

3.4.1 Erstellung beliebiger Spielstände

So ist es beispielsweise, durch die nahezu fehlende Authentifizierung der Spieler, ganz einfach möglich, durch Löschung der App(-Daten) immer wieder neue Spielstände zu erstellen. Da ein einmal verwendeter Barcode respektive die dadurch kodierte Nummer vom Server allerdings nicht mehr akzeptiert wird, muss also irgendeine andere Nummer manuell eingegeben werden. Da jedoch clientseitig alle Nummern in einem Intervall von insgesamt 60000 verschiedenen Nummern akzeptiert werden, und in der Spielrunde 2018 nur ungefähr 1000 Studenten am Spiel teilgenommen haben, ist die Wahrscheinlichkeit, innerhalb von wenigen Versuchen eine noch nicht benutzte Nummer zu finden, sehr hoch.

Um einen neuen Spielstand zu erstellen, muss nun also nur noch ein freier Nutzername eingegeben werden. Dies ist jedoch ähnlich wie bei der Eingabe einer zufälligen Barcodenummer kein Hindernis, denn zufällige Buchstaben-Zahlen-Kombinationen gibt es im Vergleich dazu wesentlich mehr.

Da bei der Registrierung das Haus, für das der neue Spielstand Punkte sammelt, frei gewählt werden kann, und zudem jeder neue Spielstand zu Beginn sofort Punkte für die Registrierung erhält, hat das gewählte Haus damit direkt einen Profit erzielt, ohne dass weitere Aktionen über diesen Spielstand erfolgen müssen. Die so künstlich erstellten Spielstände sind dabei nicht von legitimen unterscheidbar, da sie durch den normalen Registrierungsprozess erstellt wurden. Das Erkennen und Korrigieren der durch diese Vorgehensweise erschummelten Punkte ist daher sehr schwierig.

3.4.2 Time-Traveling-Bug

Als weiteres Problem ist der sogenannte „Time-Traveling-Bug“ zu nennen, an welchem auch viele andere Apps beziehungsweise Spiele leiden. Dieser Fehler bedeutet einfach gesagt, dass wenn ein Spieler die Systemzeit auf seinem Handy manuell ändert, zeitbasierte Beschränkungen im Spiel umgangen werden können. Besitzt das Spiel nämlich keinen Server, beispielsweise ein reines Offline-Spiel, oder nutzt diesen nicht um die Zeit zu verifizieren, so muss das Spiel der vom System vorgegebenen Zeit vertrauen, weil es ansonsten keine anderen Bezugsmöglichkeiten für die Zeit gibt.

Konkret bedeutet das für Game of TUK, dass die eigentlich zeitlich beschränkten Weeklies jederzeit durchgeführt werden können, also auch wenn diese ei-

gentlich bereits vorbei sind. Damit könnte ein Spieler also jederzeit verpasste Punkte noch nachholen, oder sogar dieselben Punkte nochmals sammeln, da die App nur überprüft ob sich der Tag geändert hat, nicht aber welcher Tag es eigentlich ist. In letzterem Fall könnten die Spielleiter zumindest am Zeitstempel der Punktevergabe in der Server-Datenbank erkennen, dass dieser Spieler bestimmte Punkte mehrfach eingesammelt hat und diesen daraufhin eventuell als Cheater deklarieren und aus der Wertung nehmen.

3.4.3 GPS-Spoofing

Da drei der vier Weeklies Aufgaben haben, für die der Standort respektive dessen Verlauf wichtig ist, kann ein Spieler zudem auch durch GPS-Spoofing cheaten, indem er seinen Standort fälscht. Da eine Standortangabe nicht weiter geprüft wird, beispielsweise ob zum gleichen Zeitpunkt die Einstellung die GPS-Spoofing erlaubt, aktiviert ist, muss die App also darauf vertrauen, dass der Spieler an den richtigen Standorten der Coins oder der Schnitzeljagd ist.

Dadurch kann dieser ohne Probleme an den jeweiligen Tagen innerhalb von kürzester Zeit, ohne sich dafür bewegt zu haben, alle Punkte für diese Aufgaben sammeln. Dies funktioniert natürlich ebenfalls bei den Fahrrad-Kilometern, allerdings ist hier der Aufwand wesentlich höher, da hierfür eine kontinuierliche Bewegung simuliert werden muss, oder alternativ in den richtigen Zeitabständen die richtigen Standorte.

3.4.4 Dekompilieren

Ist ein Spieler nun ein wenig mehr mit App-Entwicklung oder generell Programmieren vertraut, so ergeben sich schnell weitere Möglichkeiten zum cheaten. So kann er beispielsweise die App dekompilieren und so nicht nur weitere Bugs oder Fehler finden, sondern auch die gesamte API auslesen. Dies erlaubt ihm somit neben der Ausnutzung von Bugs auch tiefgehendere Schummelmöglichkeiten.

So hat ein Spieler, der alle Serveranfragen der App kennt, nun aufgrund der fehlenden Authentifizierung dieser Anfragen die Möglichkeit, jede beliebige Anfrage selbst zu konstruieren und zu senden. Er kann dabei einerseits, um sich selbst Punkte zu geben, legitime Anfragen nachbauen. Umgekehrt kann er aber auch einfach den Spielstand eines anderen Spieler übernehmen beziehungsweise kopieren, oder diesen sogar als Cheater darstellen, indem er in dessen Namen nicht legitime Anfragen an den Server sendet. Diese können dann einfach von den Spielleitern identifiziert werden, beispielsweise wenn eine Anfrage einem Spieler eine große Anzahl an Punkten gutschreibt, die er offensichtlich nicht auf faire Art und Weise auf einmal erhalten haben kann.

Schlussendlich kann ein Spieler mit Vollzugriff auf die Serveranfragen auch, wie bereits zu Beginn des Kapitels beschrieben, gezielt Inkonsistenzen in der Da-

tenbank hervorrufen und damit viel Schaden anrichten. Dieser Schaden kann von viel Arbeit für die Spielleitung, bis hin zum zwangsweisen Abbruch des Spiels reichen, wenn zum Beispiel fehlerhafte Daten nicht mehr unter vertretbarem Aufwand korrigiert werden können.

3.5 Fehler/Bugs

Neben den generellen Problemen gibt es auch einige Fehler in der App, die das allgemeine Spielerlebnis mindern können. So ist beispielsweise der Registrierungsprozess sehr fehleranfällig, da die Anfragen an den Server nicht gebündelt, sondern einzeln an diesen gesendet.

Beispielsweise wird zuerst die Barcode-Nummer des Studentenausweises geprüft und invalidiert. Erst danach wird der Spieler nach einem Nutzernamen gefragt. Wird die App nun aus irgendeinem Grund beendet, so startet sie beim nächsten Mal, da der Registrierungsprozess zuvor nicht vollständig abgeschlossen wurde, wieder mit diesem, allerdings ganz zu Beginn, also dort wo der Barcode eingescannt, beziehungsweise die Barcode-Nummer eingegeben werden muss. Da diese Nummer bei dem gleichen Spieler aber bereits invalidiert wurde, kann er sich nicht mehr auf regulärem Wege anmelden, außer er gibt, wie bereits in Abschnitt 3.4 beschrieben, eine beliebige Nummer an, die noch nicht benutzt wurde.

Einen weiteren Fehler gibt es bei den Quickies, denn hier sieht der Spieler eine Liste von gesendeten und empfangenen Duellen, allerdings kann er pro Minispiel jeweils immer nur das erste Duell, welches er erhalten oder selber begonnen hat, spielen. Es gibt dabei keine Möglichkeit, dieses Duell zu wechseln oder ein neues Duell gegen jemand anderes zu beginnen. Der Spieler kann lediglich das Duell abbrechen, womit er aber automatisch aufgibt und keine Punkte mehr für dieses Duell erhält. Sollte der Gegenspieler nun keinen Zug machen, eventuell weil er dies selber nicht kann, so wird der Spieler effektiv blockiert, solange er das Duell nicht aufgeben möchte. Dies führte in der Spielrunde 2018 dazu, dass viele Spieler nur noch Duelle gegen den Computer gespielt haben, da diese nicht von diesem Problem betroffen waren.

4 Restrukturierte Architektur

Um den Entwicklungsaufwand gering zu halten, wurde der Server mithilfe des Django-Frameworks [1] basierend auf Python implementiert. Dieser Server stellt hierbei, wie auch der vorherige Server, eine REST-kompatible API für den Client zur Verfügung. Dies ermöglicht insbesondere einfaches Testen des Servers, da die Testanfragen auch von kleinen Test-Clients gesendet werden können. Beispielsweise kann hierfür die Python-Bibliothek „Requests“ verwendet werden.

Während somit die neue Architektur der App serverseitig komplett neu aufgebaut wurde, basieren die Clients immer noch auf den ursprünglichen Versionen und wurden dementsprechend lediglich umgebaut, um vor allem mit dem neuen Server kompatibel zu sein. Dies lag insbesondere an Zeitgründen, allerdings konnten so auch keine weitreichenderen Änderungen oder spieltechnische Verbesserungen, wie beispielsweise andere Weeklies gemacht werden. Daher sollen zuallererst die Abwägungen erläutert werden, die im Zuge der Restrukturierung betrachtet wurden:

4.1 Zeitabwägungen und Ziele

Der hauptsächlich limitierende Faktor war, wie bereits im Jahr zuvor, die Zeit. Um den Spielzeitraum möglichst nicht in die Lernphase für den nachfolgenden Prüfungszeitraum der Studenten zu legen, war der späteste Startzeitpunkt Ende Juni 2019. Da die Implementierung erst im April desselben Jahres angefangen hat und die App planmäßig Mitte Juni testbereit sein sollte, musste die gesamte Restrukturierung innerhalb dieser zweieinhalb Monate erfolgen. Wie bereits zu Beginn des Kapitels erwähnt, wurde daher lediglich der Server komplett neu programmiert, während die Clients nur auf Basis der originalen Version modifiziert wurden. Dies war zusätzlich dadurch bedingt, dass an den Clients für Android und iOS jeweils unterschiedliche Entwickler gearbeitet haben, und daher auch gleichzeitig Absprachen getroffen werden mussten.

Um also diese zur Verfügung Zeit effizient zu nutzen, wurde sich darauf geeinigt, den Fokus auf die folgenden, insbesondere sicherheitsbezogenen Punkte zu legen:

- Neuer Registrierungs- und Login-Prozess inklusive Authentifizierung, sodass:

[1] `https://www.djangoproject.com`

  - jeder Spieler maximal einen Account haben kann
  - ein Spieler sich jederzeit wieder in diesen Account einloggen kann
  - gleichzeitig die Zugehörigkeit zur Universität überprüft wird
- Validierung, Auswertung und Speicherung der Spieldaten auf dem Server
  - Authentifizierung der Anfragen
  - Konsistenz der Daten sollte gewahrt bleiben
  - Cheating soll so weit wie möglich erschwert werden und im Zweifelsfall durch die Datenspeicherung rückverfolgbar sein
- Behebung von Berechnungs- beziehungsweise Logik-Bugs im Client
  - Diese könnten den ordentlichen Spielfluss beeinträchtigen

Dagegen wurde beispielsweise die Behebung von weniger wichtigen Fehlern wie UI-Bugs oder Textfehlern hinten angestellt, ebenso wie grundlegend neue Features, Weeklies oder Quickies, da dies für den gegebenen Zeitrahmen zu umfangreich gewesen wäre. Weil sich zudem die clientseitige Architektur auf den verschiedenen Betriebssystemen an einigen Stellen sehr stark unterscheidet, wurden zudem einige Funktionen des neuen Servers mit leicht unterschiedlichen Schnittstellen für Android und iOS bereit gestellt. Dies diente dazu, die Client-Entwicklung zu beschleunigen, da jene Stellen so nicht umständlich neu geschrieben werden mussten, sondern die schon vorhandenen Code-Elemente verwendet werden konnten.

4.2 Sicherheitsabwägungen

Weiterhin musste auch die Sicherheit abgewägt werden, insbesondere stellte sich die Frage, welchen Anteil der Berechnungen der Server übernimmt. Dabei reicht die Spannweite der Möglichkeiten von einem reinen Offline-Spiel bis hin zu einem reinen Online-Spiel. Ersteres ist natürlich alleine schon durch die Interaktionsmöglichkeiten zwischen den Spielern keine Option und könnte auch in keinster Weise irgendeine Sicherheit garantieren. Letzteres würde hingegen die Spielfreiheit einschränken und könnte insbesondere auch Spieler des Spiel ausschließen, denn Spieler die beispielsweise auf WLAN angewiesen sind, könnten an keinem der Weeklies, die außerhalb des Campus der Universität stattfinden, teilnehmen, sofern eine permanente Internetverbindung erforderlich wäre. Außerdem wäre eine solche Variante datenschutzrechtlich sehr schwierig, da hierfür alle Sensordaten des Handys auf den Server übertragen werden würden, also insbesondere auch Standortverlauf und Ähnliches.

Aus den genannten Gründen wurde daher eine Variante, die sich ungefähr in der Mitte zwischen den beiden Extremen befindet, gewählt. So soll der Spieler alle Weeklies, wie auch in der Spielrunde 2018, prinzipiell komplett offline spielen können. Dabei werden die relevanten Daten in erster Instanz lokal auf

dem Handy des Spielers ausgewertet. Sofern eine Aktion, beispielsweise das Einsammeln einer Coin während des Coin-Collector-Weekly, lokal gültig ist, wird eine dazugehörige Serveranfrage in einem lokalen Jobmanager gespeichert. Sobald das Handy des Spielers wieder eine Verbindung zum Internet aufgebaut hat, werden diese Anfragen nacheinander ausgeführt, also an den Server gesendet. Die Daten werden dort dann nochmal in zweiter Instanz überprüft und falls die Aktion tatsächlich gültig ist, wird die vollführte Aktion auf dem Server gespeichert und der Spieler erhält final die zugehörige Punktzahl.

Diese Variante hat den Vorteil, dass sowohl Sicherheit gewährleistet ist, da alle Aktionen vom Server überprüft werden und daher die Konsistenz der Datenbank gesichert werden kann, als auch die Nutzerfreundlichkeit erhalten bleibt, da die wichtigen Spielelemente ohne aktive Internetverbindung gespielt werden können.

Schlussendlich ergibt sich durch diese Verfahrensweise aber auch, dass einzelne Sicherheitsaspekte, im Gegenzug für besseren Datenschutz und Nutzerfreundlichkeit nicht angewendet werden können. So ist zum Beispiel eine Validierung in Echtzeit, beispielsweise die permanente Überprüfung von Geodaten auf dem Server nicht machbar. Daher müssen unter anderem auch, damit eine lokale Überprüfung möglich ist, die echten Koordinaten der Coins und Schnitzeljagdorte der App bekannt sein. In der neuen Implementierung werden diese daher bei Bedarf dynamisch vom Server geladen. Dadurch müssen diese Standorte nicht von Vorneherein festgelegt werden. Zudem sind diese somit nicht fest im Quellcode einprogrammiert, sondern werden nach Abruf im internen Speicher gespeichert. Technisch versierte Nutzer haben aber dennoch die Möglichkeit, diese aus der App auszulesen, allerdings benötigt dies im Vergleich mehr Aufwand.

4.3 Django-Framework

Django ist ein Web-Framework das lose auf dem Model-View-Controller (MVC) Muster basiert. [Ree79] Dieses Muster unterteilt dabei den Code in:

Models welche die Daten enthalten

Views die definieren wie diese Models dargestellt beziehungsweise präsentiert werden

Controller die Programmlogik, die sowohl die Models als auch die Views verwaltet, und diese bei Bedarf, beispielsweise bei Nutzerinteraktion, modifizieren kann

Die Schnittstelle, mit der Nutzer interagieren, sind dabei die Views.

Tatsächlich nutzt Django laut der offiziellen Webseite [19a] allerdings eine leicht modifizierte Variante, nämlich das Model-View-Template (MVT) Muster, welches wie folgt aufgebaut ist:

Views Im Gegensatz zum MVC Muster sind die Views hier lediglich als serverseitige Repräsentation der Models zu verstehen. Gleichzeitig definieren diese aber auch Elemente des Controllers und sind somit die direkte Schnittstelle zu den Templates

Templates Diese definieren was der Nutzer auf der Webseite tatsächlich sieht und bilden daher also die Views des MVC Musters ab. Sie besitzen dabei Funktionalitäten um mit den Views des Servers zu kommunizieren, beispielsweise durch ein Formular, welches Nutzer ausfüllen und an den Server senden können.

Models Wie beim MVC Muster enthalten Models die Daten und sind mit den Server-Views sowie einer meist zugrundeliegenden Datenbank gekoppelt.

Dies bedeutet insbesondere, dass durch die Verwendung von Models keine direkte Interaktion mit der Datenbank vonnöten ist, da diese indirekt durch die Views, welche unter anderem die Models verwalten, durchgeführt wird.

Django ist daher für die geplante Server-Restrukturierung von Vorteil, da dieses Muster ideale Voraussetzungen für eine REST-kompatible API besitzt. So können wir die Views als API-Schnittstellen verwenden, während die Models die nötigen Daten für das Spiel beinhalten. Jede View beschreibt dabei eine bestimmte Funktionalität. Daher kann eine HTTP-Request an eine View, als Aufruf einer Funktion des Servers mit den in der Request gegebenen Parametern angesehen werden. Fehlen dabei Parameter, so kann die View eine HTTP-Response mit einem gegebenen Fehlercode zurückgeben, ohne die Funktion auszuführen. Im weiteren Verlauf werden daher alle Views als parametrisierter Funktionsaufruf repräsentiert, dies erleichtert die Erklärung. Da fast alle Views API-Elemente sind und dementsprechend nur JSON-Strings zurück geben, sind zudem prinzipiell keine Templates nötig. Lediglich eine einzige View, die dem Spieler während der Anmeldung einen Key anzeigen muss, besitzt daher ein Template. Ansonsten findet die Kommunikation mit dem Server nur über die App statt.

Um nun diese Server-Funktionen erklären zu können, werden im folgenden die Models beschrieben, welche zur Datenhaltung verwendet werden:

Model	Beschreibung
AuthUser	Enthält für jeden Account die zur Authentifizierung nötigen Elemente
User	Verwaltet den Spielstand jedes Accounts
EventLog	Jedes EventLog enthält die Informationen einer bestimmte Aktion eines Spielers zu einer bestimmten Zeit, inklusive der erhaltenen Punkte
Duels	Verwaltet die Duelle beider Arten durch eindeutige Ids
DonkeyDuels	Enthält alle Spieldaten für das gegebene Duell
RPSDuels	Enthält alle Spieldaten für das gegebene Duell
QRCode	Verwaltet alle QR-Codes des Sportkurs-Weekly
HousePoints	Jeder Eintrag enthält den Punktestand aller Häuser zu einer bestimmten Zeit
Riddle	Verwaltet alle Schnitzeljagd-Rätsel
Coin	Verwaltet alle Coins
SuperCoin	Verwaltet alle Supercoins inklusive ihrer Gültigkeitstage
News	Verwaltet alle Neuigkeiten

4.4 Registrierung/Login

Da das Registrierungs- und Login-Verfahren in der Version von 2018 eine der größten Schwachstellen des Spiels ist, wurde dieses neu konzipiert und mithilfe des Authentifizierungsverfahrens „Shibboleth“ realisiert. Dieses neue Verfahren ermöglicht dabei die Identifizierung des Spielers, sodass es diesem möglich ist, jederzeit wieder Zugang zu seinem Spielstand zu erhalten, sollte er diesen verloren haben. Dabei werden auf dem Server von Game of TUK selbst keine Identifikationsmerkmale des Spielers gespeichert, da die eigentliche Identifizierung von Shibboleth übernommen wird. Dies ist für die Spieler ein gutes Merkmal für den Datenschutz, da sich ihr Spielaccount inklusive potenzieller Bewegungsprofile nicht auf sie selbst zurückverfolgen beziehungsweise zuordnen lässt.

4.4.1 Shibboleth

Shibboleth[2] ist ein Verfahren zur Authentifizierung, dass unter anderem bei Webservices der TU Kaiserslautern verwendet wird, um autorisierten Nutzern den Zugang zu diesen Diensten zu ermöglichen. So benutzt beispielsweise die Lernplattform OpenOLAT, mit der viele der an der Universität angebotenen Vorlesungen verwaltet werden, ebenfalls Shibboleth zur Authentifizierung der Nutzer. Um diese Dienste also nutzen zu können, hat jeder Angehörige der Universität eine Zugangskennung beim sogenannten Identity-Provider, welcher in diesem Fall vom „Regionalen Hochschulrechenzentrum Kaiserslautern“ bereitgestellt wird.

[2] `https://www.shibboleth.net`

Möchte ein Nutzer Zugriff auf einen bestimmten durch Shibboleth geschützten Dienst haben – dieser wird von einem Service-Provider bereitgestellt –, so wird der Nutzer vom Service-Provider zum Identity Provider weitergeleitet. Dieser authentifiziert den Nutzer und schickt diesen zurück zum Service Provider. Der Aufruf beim Service Provider erfolgt mit einem Parameter, der den Authentifizierungsvorgang kennzeichnet. Über diesen Parameter kann der Service Provider dann beim Identity Provider die Authentifizierung nachvollziehen und zuvor vereinbarte Daten über den Nutzer abrufen. Anhand dieser Daten, wie beispielsweise dem Studentenstatus oder der Fachbereichszugehörigkeit, entscheidet der Service-Provider dann, ob der Nutzer berechtigt ist, Zugriff zu diesem Dienst zu erhalten. Ist dies der Fall, so kann der Nutzer den Dienst eine begrenzte Zeit lang nutzen. Nach Ablauf der Zeit muss er sich erneut authentifizieren.

Dabei erfährt der Service-Provider über den Nutzer nur die Daten, die zur Nutzung des Dienstes erforderlich sind. Jene Daten sind durch die Registrierung des Dienstes bei Shibboleth festgelegt und werden dem Nutzer vor der erstmaligen Nutzung des Dienstes auf Wunsch angezeigt. Da insbesondere nur dann persönliche Daten wie der echte Name des Nutzers übertragen werden, wenn dies für den Dienst unbedingt erforderlich ist, kann somit sehr einfach Datenschutz garantiert werden.

Durch die häufige Nutzung von Shibboleth an der Universität ist jene Authentifikationsprozedur den Studenten, also der Hauptzielgruppe von Game of TUK, sehr gut bekannt und vertrauenswürdig. Durch die gleichzeitig erfolgende Identifikation als Universitäts-Angehöriger ist Shibboleth daher ideal geeignet, um bei der Registrierung für Game of TUK verwendet zu werden. Dabei ist das einzige Datum, das Game of TUK benötigt und daher abfragt, eine Identifikationskennung. Diese ist hierbei für jeden Nutzer bei jedem unterschiedlichen Dienst einzigartig und wird vom Identity-Provider eindeutig festgelegt. Dadurch kann ein einzelner Dienst einen seiner Nutzer immer wieder erkennen, aber zwei verschiedene Dienste können keinerlei Nutzerdaten oder Profile miteinander in Verbindung bringen.

4.4.2 Anmelde-Prozess

Client

Sobald ein Spieler die App zum ersten Mal startet, werden ihm die Datenschutzerklärung und die Nutzungsbedingungen angezeigt, welchen der Spieler zustimmen muss. Nachdem er im nächsten Screen zusätzlich bestätigt, dass er verspricht nicht zu cheaten, wird er aufgefordert einen Key einzugeben. Um diesen zu erhalten, wird er via Button auf eine durch Shibboleth geschützte Webseite geleitet. Nach Eingabe seiner Zugangskennung – dies passiert dabei auf einer dedizierten Seite des Identity-Providers und nicht auf einer vom Game of TUK Server kontrollierten Webseite – wird dem Nutzer ein 8-stelliger

Key bestehend aus Zahlen und Buchstaben angezeigt. Diesen, zwei Stunden lang gültigen Key, muss er kopieren und in der App in das dafür vorgesehene Feld einfügen. Nach Bestätigung und serverseitiger Überprüfung des Keys muss er dann noch seinen Fachbereich, für den er spielen möchte, sowie einen Nutzernamen und falls gewünscht eine E-Mail zur Umfrageteilnahme angeben. Besitzt bereits ein anderer Spieler den gewählten Nutzernamen, so muss ein neuer Name gewählt werden. Sobald ein noch nicht verwendeter Name eingegeben wurde, wird der Account final angelegt und der Spieler wird zum eigentlichen Hauptmenü (siehe Abbildung 2.1) weitergeleitet. Dies ist ab dann die Startseite der App.

Hat ein Spieler bereits einen Account und möchte sich neu einloggen, so muss er bis zur Eingabe des Keys dieselbe Prozedur durchlaufen. Gibt er nun allerdings seinen Key – dieser kann durchaus ein anderer als der initial verwendete sein – ein, so findet der Server den mit dem Spieler verknüpften Account und der Spieler wird daher sofort zum Hauptmenü weitergeleitet.

Server

Auf dem Server laufen parallel dazu die folgenden Prozesse ab:

Sobald Shibboleth den Spieler authentifiziert hat, wird dieser auf die geschützte Webseite, die den Key generiert, weitergeleitet. Dabei überträgt Shibboleth im Hintergrund eine eindeutige Id an den Server, welche aber bei der Authentisierung desselben Spieler ebenfalls immer gleich ist. Diese wird auf dem Server im Model AuthUser permanent gespeichert und kann später dazu verwendet werden, den Account des Spielers zu finden und wiederherzustellen, falls dieser beispielsweise die App deinstalliert. Zum gleichen Zeitpunkt wird auch der Key zufällig generiert und gespeichert, ebenso wie sein Ablaufdatum.

Gibt der Spieler nun diesen Key in der App ein, so wird die Funktion **checkTmpKey()** des Servers aufgerufen und überprüft die Existenz und Gültigkeit des Keys. Enthält das dazu gehörige AuthUser-Model einen Nutzernamen, so hat der Spieler bereits einen Account mit ebendiesem Namen. Dieser Name wird dann zusammen mit einem Secret – Details hierzu folgen im Unterkapitel 4.5.1 – an den Client zurückgesendet, welcher den Spieler dann in seinen Account einloggt. Ansonsten wird lediglich die Gültigkeit bestätigt und der clientseitige Registrierungsprozess wird initiiert.

Sobald der Spieler in der Registrierung einen Nutzernamen gewählt hat, so wird die Server-Funktion **register()** aufgerufen. Dabei überprüft der Server zuerst nochmals den Key auf Gültigkeit, da dieser in der Zwischenzeit abgelaufen sein könnte. Dies dient dabei zusätzlich der Sicherheit, da so auch ein nachgebauter Client, der den vorherigen Key-Check umgeht, einen Spieler nicht ohne gültigen Key registrieren kann.

Erst danach wird geprüft ob der eingegebene Nutzernamen schon verwendet wird. Ist dies nicht der Fall, so wird der Account final registriert. Dies be-

deutet, dass ein zugehöriges User-Model erstellt wird, welches im folgenden Spielverlauf die Punkte und Duelle verwaltet. Das AuthUser-Model hingegen enthält als Referenz lediglich den Nutzernamen, um die beiden Modelle in Verbindung zu bringen, allerdings ist diese Referenz eine reine Namensnennung anstatt eines Fremdschlüssels. Dies ermöglicht es, dass ein Spieler nach Bedarf seinen Account – gegeben durch das AuthUser-Model – auf Nachfrage löschen lassen kann, ohne dass der Spielstand – gegeben durch das User-Model – gelöscht werden muss, denn eine solche Löschung würde Inkonsistenzen in der Datenbank hervorrufen, da beispielsweise alle zu dem Spielstand gehörigen Duelle die nötige Referenz auf einen der Duellpartner verlieren würden. Schlussendlich erhält der Client in jedem Fall die Bestätigung zur Account und Spielstand-Erstellung, und dazu, wie beim Login, ein Secret, das für die Authentifizierung aller weiteren Anfragen verwendet wird.

In jedem der Fälle hat der Client also zu irgendeinem Zeitpunkt ein Secret erhalten. Daraufhin sendet der Client eine Bestätigung an den Server, dass er dieses erhalten hat, und um sicherzustellen, dass dieses auch korrekt ist. Dies wird durch die **ackSecret()** Funktion überprüft. Ist das Secret korrekt, so kann das Spiel ab sofort normal gespielt werden. Ist dies nicht der Fall, gab es vermutlich einen Fehler bei der Datenübertragung oder Ähnliches. Dann wird ein neues Secret gesendet und der Bestätigungsprozess wiederholt sich.

4.5 Serverseitige Kontrollmechanismen

Neben den bereits erwähnten Neuerungen beim Anmeldeprozess wurde auch der restliche Teil des Servers neu konzipiert. Dadurch konnten serverseitige Kontrollmechanismen eingeführt werden, die Anfragen authentifizieren und validieren. Im Folgenden werden daher die dazu nötigen Prozesse beschrieben. Diese umfassen eine Prüfung der Urheberschaft einer Anfrage. Ebenso wird die Validität im Sinne der Spiellogik sichergestellt. So ist insbesondere auch die Datenspeicherung aller Aktionen der Spieler ein Hauptbestandteil dieser Prozesse, da so geprüft werden kann, welcher Spieler noch welche dieser Aktionen ausführen kann.

4.5.1 Absenderprüfung der Anfragen

Wie bereits zuvor erwähnt, erhält der Client vom Server während des Anmeldeprozesses ein zufälliges, 256-stelliges Secret bestehend aus Buchstaben und Zahlen. Dass dieses korrekt erhalten wurde, wird durch den erfolgreichen Aufruf von **ackSecret()** bestätigt, andernfalls wird ein neues Secret gesendet. Sobald also beide Seiten dieses Secret bestätigt haben, wird es sowohl auf dem Server als auch auf dem Client mit SHA-256 gehasht und final gespeichert. Da nun unter der Annahme, dass niemand diese Secret-Übertragung mitgehört hat, nur der Server und dieser eine spezifische Client dieses Secret kennen,

kann dieses als Zertifikat für jede weitere Anfrage verwendet werden. Dies funktioniert wie folgt:

1. Der Client möchte eine Anfrage mit k Parametern an den Server senden.
2. Da die Liste der Parameter geordnet ist, können diese k Parameter zu einem eindeutigen String *str* konkateniert werden.
3. Zusätzlich wird noch das Secret des Client mit dem String *str* konkateniert, also: $secretstr = Secret + str$
4. Dieser String *secretstr* wird mit SHA-256 gehasht und erzeugt somit den Hash $h = \text{SHA-256}(secretstr)$
5. Der Client sendet nun die Anfrage an den Server, allerdings enthält diese Anfrage neben den ursprünglichen k Parametern auch noch den Parameter $hash = h$
6. Da auch der Server das Secret des Client kennt, kann er die übergebenen Parameter – ohne den zusätzlichen hash-Parameter – ebenfalls zusammen mit dem gespeicherten Secret konkatenieren und mit SHA-256 hashen.
7. Entspricht das so berechnete Ergebnis dem des übertragenen hash-Parameter, so ist die Anfrage valide und die angefragte Funktion wird mit den gegebenen Parametern ausgeführt.

Nach der Konstruktion von SHA-256 würde offensichtlich jede kleinste Manipulation an einem der Anfrageparameter dazu führen, dass der Hash nicht mehr korrekt wäre und die Anfrage vom Server abgelehnt werden würde. Somit kann also ohne Kenntnis des Secrets keine manipulierte Anfrage an den Server gestellt werden. Umgekehrt kann aus einer Anfrage auch nicht das Secret ausgelesen oder berechnet werden. Dies liegt daran, dass SHA-256 als kryptographische Hash-Funktion eine sogenannte Einwegsfunktion ist, also nicht zurückberechnet werden kann.

Hätte ein Angreifer nun auf irgendeine Art und Weise eine Anfrage mitgehört und würde somit dessen Parameter kennen, so hätte er theoretisch die Möglichkeit diese Anfrage einfach erneut zu senden. Ein solcher Angriff, der Replay-Attacke genannt wird, würde aber nicht funktionieren, beziehungsweise hätte keinen gewinnbringenden Effekt. Dies wird in Kapitel 5.2 genauer erklärt.

4.5.2 Validität von Anfragen

Die neuen Serverkomponenten prüfen neben Authentizität auch die Plausibilität der Anfragen. So wird insbesondere auch die Validität von Anfragen, im speziellen diejenigen einzelner Aktionen, serverseitig nochmals überprüft. Beispielsweise wird jede Anfrage, die einen timestamp-Parameter besitzt, auf

Aktualität geprüft, das bedeutet dass geprüft wird, ob die durch den Zeitstempel angegebene Zeit nicht in der Zukunft liegt, beziehungsweise nicht zu weit, um auch auf durch ungenaue Handy-Uhren oder eine falsch eingestellte Zeitzone entstandene Differenzen Rücksicht zu nehmen. Für alle Weeklies wird weiterhin überprüft, ob der jeweilige Zeitstempel tatsächlich auch innerhalb des Spielzeitraums des Weekly liegt. Ansonsten werden für die Weeklies, die folgenden, speziellen Validierungen ausgeführt:

- Coin-Collector

  coinCollected() Neben Nutzername, Zeitstempel und Hash werden hier insbesondere Koordinaten und eine Coin-Id übergeben. Der Server überprüft nun, neben den allgemeinen Authentizitäts- und Zeit-Prüfungen, ob eine Coin mit der gegebenen Id existiert, ob der Nutzer diese nicht schon eingesammelt hat, und ob er nah genug an der genannten Coin war. Hierfür wird die Weg-Differenz zwischen den gegebenen Koordinaten und dem echten Standort der Coin, welche im zugehörigen Coin-Model gespeichert ist, berechnet. Dabei wird eine größere Toleranz als auf dem Client gewährt, damit nicht durch unterschiedliche Rundungen auf Client und Server eine eigentlich valide Anfrage abgelehnt wird. Ist die Anfrage valide, so wird ein EventLog für diese Aktion erstellt und der Spieler erhält somit final die Punkte dafür. Dass hierbei die Koordinaten nochmal auf dem Server überprüft werden, erschwert auch das Cheating, denn sonst könnte ein Spieler die App nach Dekompilierung so umbauen, dass die clientseitige Überprüfung des korrekten Standorts einfach übersprungen wird. Auf diese Weise müsste er die korrekten Koordinaten zuerst herausfinden, bevor er manipulierte Anfragen senden kann.

  getSuperCoin() Da der Supercoin erst erhalten werden kann, wenn der Spieler alle normalen Coins an diesem Tag gesammelt hat, wird in dieser Funktion vor Ausgabe des zugehörigen Rätsels geprüft, ob der Spieler dies tatsächlich geschafft hat. Zusätzlich wird auch noch geprüft, ob er die jeweilige Supercoin nicht bereits eingesammelt hat, weil in diesem Fall kann er dies natürlich nicht nochmal tun und erhält die entsprechende Rückmeldung.

  superCoinCollected() Die Valdierung läuft hier ähnlich wie bei einer normalen Coin ab, nur muss hier keine Supercoin-Id angegeben werden, da es pro Tag nur eine Supercoin gibt. Ansonsten wird auch hier nochmal überprüft, dass der Spieler an dem Tag alle normalen, aber noch nicht die Supercoin eingesammelt hat.

- Fahrrad-Kilometer

  metersDriven() Der Server erhält Start- und End-Zeit von einem 100 Meter Abschnitt sowie drei Gyro-Sensor-Werte. Letztere sollten ursprünglich verwendet werden um festzustellen, ob sich das Handy

des Spielers tatsächlich bewegt. Eine solche Überprüfung wurde allerdings nicht implementiert. Dies lag dabei am zu hohen Aufwand einer experimentellen Ermittlung realistischer, zulässiger Werte. Stattdessen wird lediglich überprüft, ob die Geschwindigkeit des Spielers – diese ist berechenbar durch die Zeitdifferenz – innerhalb eines Intervalls liegt, das für Fahrradfahren üblich ist. War der Spieler zu langsam oder zu schnell, wird dies dem Client zurückgemeldet, andernfalls erhält der Spieler, sofern er noch nicht mehr als die tägliche Maximaldistanz zurückgelegt hat, hierfür die Punkte durch einen EventLog-Eintrag. Weiterhin werden zum gleichen Zeitpunkt auch noch Bonuspunkte für täglich erreichte Meilensteine, beispielsweise für fünf zurückgelegte Kilometer vergeben.

- Schnitzeljagd

  nodeVisited() Wie beim Coin-Collector-Weekly werden auch hier Koordinaten und eine Rätsel-Id übergeben. Falls das angegebene Rätsel existiert, so wird auch hier überprüft ob der Spieler dieses schon gelöst und Punkte dafür erhalten hat. Ist dies nicht der Fall und sind die gegebenen Koordinaten nah genug an des Rätsels Lösungs-Ort, so erhält der Spieler die dafür ausgeschriebenen Punkte. Zusätzlich erhält er, sofern er bestimmte Teile der Schnitzeljagd erfolgreich abgeschlossen hat, auch noch Bonuspunkte.

- Sportkurse

  addPointsForCourse() Der Server muss hierbei lediglich überprüfen, ob der übergebene Code des auf dem Client eingelesenen QR-Code valide ist, also insbesondere existiert und noch nicht benutzt wurde. Ist dies der Fall, so wird der Code gleichzeitig invalidiert und der Spieler erhält die Punkte.

Generell muss dazu natürlich noch erwähnt werden, dass einigen Daten wie beispielsweise den übergebenen Koordinaten geglaubt werden muss, da Sensordaten im Allgemeinen niemals mit hundertprozentiger Sicherheit auf einem Server geprüft werden können. Es können lediglich clientseitig Hindernisse eingebaut werden, die eine Manipulation von diesen Daten ohne technisches Hintergrundwissen erschweren.

Werden nun im Gegensatz dazu die Quickies betrachtet, so fällt auf, dass diese komplett auf dem Server berechnet werden; der Client agiert hierbei nur als Input-Output-Schnittstelle. So kann der Client lediglich ein Duell anfragen: **sendChallenge()**, ein solches Duell akzeptieren oder ablehnen: **acceptChallenge()** oder **refuseChallenge()** und natürlich die Züge des Spielers an den Server übermitteln, also mit **sendTime()** die vom Spieler erzielte Zeit beim Eselsrennen oder mit **sendChoice()** die gewählte Aktion bei Schere-Stein-Papier. Der Server ermittelt dann aus den gegebenen Daten jeweils den Status der Duelle und vergibt, sobald ein Spieler gewinnt, automatisch die jeweiligen Punkte an die Spieler.

4.5.3 Datenspeicherung

Im Gegensatz zum vorherigen Server wurde dieses Mal auch Wert darauf gelegt, eine stabile und vollständige Datenhaltung zu haben, die es der Spielleitung ermöglicht, einen Überblick über alle Geschehnisse zu bewahren. Das Kernstück dieser Datenhaltung sind dabei die EventLogs, die für jede erfolgreiche, zu bepunktende Aktion eines Spieler angelegt werden. Dabei enthält jedes Eventlog die folgenden Elemente:

user Ein Fremdschlüssel auf das zugehörige User-Model

event Das Weekly, Event, Umfrage oder Duell für das der Spieler Punkte erhält

action Die genaue Aktion, also beispielsweise ob ein Duell gewonnen oder verloren wurde

points Die Punkte die der Spieler hierfür erhält

ts Der Zeitstempel zu dem die Aktion erfolgt ist

value Optional: Kann einen zur Aktion gehörigen Wert enthalten, wie beispielsweise die Id der gesammelten Coin

Damit ist sichergestellt, dass jeder Punkt den ein Spieler erhält, auch mit einer Aktion verbunden ist, die nachvollzogen werden kann. Sollte ein Spieler also beispielsweise unnatürlich viele Punkte haben, so muss es dafür zugehörige EventLogs geben, die dann geprüft werden können. So können letztendlich auch sonstige fehlerhafte Einträge, einfach gefunden, analysiert und korrigiert werden. Dies kann sogar schon in der Testphase genutzt werden, da Funktionen die fehlerhafte EventLogs generieren oder unter falschen Bedingungen, so zum Beispiel wenn ein Weekly noch nicht gestartet ist und dennoch durch eine dementsprechend ungültige Aktion ein gültiges EventLog erzeugt wird, dies durch ebendiese Erstellung offenlegen.

Wichtig zu erwähnen ist hierbei auch, dass für einige der Weeklies auch genaue Standortdaten an den Server geschickt werden, die jedoch nur für die Validierung verwendet und danach an keiner Stelle auf dem Server gespeichert werden. Zudem erhält der Server auch nur dann Standortdaten, wenn der Spieler dies explizit auf Knopfdruck bestätigt, siehe Unterkapitel 4.6.1

Weiterhin wurden viele der für die Weeklies wichtigen Daten nicht mehr fest in die App einprogrammiert, sondern sobald sie benötigt werden, dynamisch vom Server geladen. Da diese Daten somit vom Server gespeichert und verwaltet werden, ist es auch nach Veröffentlichung der App möglich, diese Daten zu verändern, ohne dass Spieler eine neue Version der App herunterladen müssen. So werden unter anderem alle (Super-)Coins mit ihren Standorten, alle Schnitzeljagd-Rätsel mit ihren Standorten, sowie natürlich auch alle Duelle mit dem jeweiligen Status serverseitig gespeichert.

Schlussendlich führen die genannten Punkte auch dazu, dass die Punktebe-

rechnung komplett auf dem Server möglich ist. Hierbei muss für jeden Spieler lediglich die Summe der Punkte aller zu seinem User-Model gehörigen Event-Logs berechnet werden. Daraus lassen sich dann die Punktestände der Häuser berechnen, indem die Punkte aller Spieler des jeweiligen Hauses akkumuliert werden und eventuelle Haus-Bonuspunkte, die in Events erspielt wurden, dazugerechnet werden.

4.6 Verbesserter Client

Im Zuge der Server-Restrukturierung musste natürlich auch der Client angepasst werden. Dabei war das primäre Ziel, die Schnittstelle zum Server herzustellen, sodass die Spielelemente, die zuvor schon vorhanden waren, vollständig funktionieren. Erst danach wurden einige neue Spielelemente zu den bestehenden Weeklies hinzugefügt beziehungsweise Funktionalitäten modifiziert, die das Spielvergnügen erhöhen. Da der Server von 2018 ebenfalls auf einer REST-kompatiblen API aufbaut, war die Anbindung an diesen mit nur wenig Aufbau verbunden. Um hingegen die neuen Server-Funktionen beziehungsweise die neu spezifizierten Schnittstellen ansprechen zu können, musste die zugrundeliegende Architektur an einigen Stellen stärker modifiziert werden als an anderen.

Dies betrifft insbesondere den Registrierungs- und Login-Prozess, da dieser im Zuge der Sicherheitsverbesserungen neu designt wurde. Weiterhin mussten die Weeklies so verändert werden, dass die gesammelten Daten nicht nur lokal verarbeitet werden, sondern zusätzlich auch noch an den Server zur erneuten Validierung gesendet werden. Da dadurch auch einzelne Anfragen im Nachhinein noch als nicht gültig deklariert werden können, musste der Client so angepasst werden, dass er die dazugehörigen invaliden Aktionen erneut verfügbar macht, denn diese wurden ja vom Server als noch nicht erfolgreich erledigt markiert. Wie dies letztendlich für die Spielrunde 2019 umgesetzt wurde, insbesondere im Bezug auf die Differenzen zwischen Android und iOS, wird im Kapitel 6 näher erläutert.

4.6.1 Anpassung/Verbesserung der Spielelemente

Neben den Anpassungen um den Client server-konform zu machen, wurden auch einige Modifikationen an den einzelnen Spielelementen vorgenommen, um die allgemeine Spielerfahrung zu verbessern. Eines der Ziele war dabei, trotz der geringen Zeit „Neues“ in die App zu bringen, um insbesondere die Spieler aus 2018 erneut für das Spiel zu begeistern. So wurden beispielsweise die folgenden Elemente hinzugefügt:

Coin-Collector Die Positionen der Coins wurden verändert, zudem wurde die Supercoin eingeführt, welche erst eingesammelt werden kann, wenn zuvor alle anderen normalen

Coins gesammelt wurden. Dabei ist diese nicht auf der Karte verzeichnet, sondern nur durch ein Bilder-Rätsel gegeben, welches sich jeden Tag ändert. Die Spieler erhalten die Supercoin, wenn sie den auf dem Bild dargestellten Ort finden und aufsuchen.

Fahrrad-Kilometer Der Algorithmus zur Berechnung der Distanz wurde leicht modifiziert um weniger false-negatives zu erzeugen. Dies bedeutet dass es weniger Fälle geben soll, in denen eine gültig gefahrene Strecke fälschlicherweise als ungültig erkannt wird. Außerdem wurden Meilensteine hinzugefügt, die bei Erreichen bestimmter zurückgelegter Distanzen Bonuspunkte geben.

Schnitzeljagd Die Schnitzeljagd-Rätsel wurden verändert, zudem wurde die Anzahl der Rätsel verdoppelt und die Schnitzeljagd in zwei Abschnitte geteilt: Wald-Schnitzeljagd und Stadt-Schnitzeljagd, die nacheinander absolviert werden können und bei Abschluss jeweils Bonuspunkte geben.

Sportkurse Hier wurde nicht die App an sich verändert, jedoch wurde der Zeitraum in dem die Spieler Game of TUK Geldscheine sammeln können auf eine ganze Woche verlängert, sodass es in allen Kursen innerhalb dieser Woche die Möglichkeit gab, Geldscheine zu erhalten.

Als komplett neue Funktion ist hingegen ein News-System dazugekommen. Dieses ermöglicht, im Gegensatz zu fest eingebauten Nachrichten zu bestimmten Zeiten, dynamisch Nachrichten, so wie sie benötigt werden, in die App zu laden. Dabei können unter anderem spontane Ankündigungen oder auch aufgetretene Probleme direkt an die Spieler weitergeleitet werden, ohne einen Umweg beispielsweise über eine Webseite zu gehen. Zudem ist so die Wahrscheinlichkeit wesentlich höher, dass Spieler diese Neuigkeiten auch mitbekommen, da diese auch über Notifications außerhalb der App angezeigt werden.

Problembehebung

Neben diesen neuen Spielelementen wurden allerdings auch noch einige der in Kapitel 3 genannten Probleme behoben. So wurde vor allem der stellenweise problematische Datenschutz angegangen. Insbesondere wurde in der neuen Client-Version beim Coin-Collector Weekly auf die Nutzung von Google Maps beziehungsweise Apple Maps verzichtet. Stattdessen werden nun die Karten von OpenStreetMap verwendet (siehe 2.2), da diese keinerlei Standortdaten an irgendeinen dritten Server senden. Zudem wird der Standort des Spielers nur noch dann ermittelt, wenn er auf den dazugehörigen Knopf drückt.

In Unterkapitel 4.5.2 wurde beschrieben, dass zur Validierung der Anfrage

auch Standortdaten an den Server gesendet werden, dies geschieht jedoch nur dann, wenn der Spieler nach der clientseitigen Überprüfung auch tatsächlich eine Coin eingesammelt beziehungsweise bei der Schnitzeljagd den richtigen Ort aufgesucht hat. Zudem werden diese Daten wie in 4.5.3 erwähnt, auch nicht auf dem Server gespeichert, sondern nur zur einmaligen Validierung verwendet.

Zu guter Letzt wurde zudem auch noch, leider aus Zeitgründen nur auf Android, der Blockier-Fehler bei den Quickies behoben. Dabei kann der Spieler nun das aktive Duell zurückstellen, oder auf Knopfdruck auch zu einem anderen Duell wechseln wenn er möchte. Dies ermöglicht insbesondere auch, dass beliebig viele weitere Gegner herausgefordert werden können, ohne dass der Spieler irgendein bestimmtes Duell zuerst beenden muss.

5 Angriffsszenarios

Um die Sicherheit des neu kreierten Servers zu testen, sollen im folgenden einige Angriffsmethoden ausprobiert werden, die diese Sicherheit aushebeln könnten. So würde sich ein potentieller Angreifer durch einen Man-in-the-Middle Angriff beispielsweise erhoffen, Spieldetails von anderen Mitspielern zu erfahren oder deren Anfragen zu manipulieren. Im Falle der Dekompilierung der App wäre es das Ziel des Angreifers, die App-Logik herauszufinden, um damit beispielsweise gefälschte Anfragen an den Server zu senden oder gefundene Schwachstellen zu nutzen, um im Spiel zu cheaten. Ansonsten gibt es noch die „klassischen“ Web-Angriffe, die im Folgenden betrachtet werden.

5.1 Relevanz klassischer Web-Angriffe

Unter diese Kategorie zählen Angriffe, die beispielsweise die Kompromittierung des Servers oder das Ausspähen von Nutzerdaten unerfahrener Nutzer zum Ziel haben und um dies zu erreichen, Schwachstellen des Browsers oder direkt des Servers ausnutzen. An dieser Stelle wären also unter anderem Cross-Site-Scripting, Cross-Site-Request-Forgery, SQL-Injection, (Distributed-)Denial-of-Service, Phishing sowie auch Man-in-the-Middle Angriffe zu nennen, wobei letzterer gesondert im nächsten Abschnitt behandelt wird. Ansonsten besitzen die genannten Angriffe aber für Game of TUK nur eine sehr geringe Relevanz.

Das liegt unter anderem daran, dass einige dieser Angriffe das unbedachte Klicken auf Links beziehungsweise das dahinterliegende Aufrufen einer Webseite ausnutzen. Allerdings dient der Game of TUK Server hauptsächlich als API und nicht als Webseite. Dies betrifft vor allem Cross-Site-Scripting sowie Cross-Site-Request-Forgery. Diese zielen hierbei darauf ab, im Browser des Opfers beispielsweise Aktionen auszuführen oder Webseiten aufzurufen, die der Nutzer nicht intendiert hat. So kann beispielsweise bei Cross-Site-Scripting ein injiziertes Script die Session eines Nutzers einer bestimmten Webseite als Cookie an den Angreifer schicken. Dieser kann nun diese Session übernehmen und damit beispielsweise Käufe tätigen oder Ähnliches. Bei einer Cross-Site-Request-Forgery wird hingegen die konkrete Aktion in einer gefälschten URL enkodiert und falls das Opfer auf der zur URL gehörenden Seite angemeldet ist, diese Aktion ausgeführt. Dies funktioniert, da der Nutzer durch seine aktive Session zur Ausführung dieser Aktion autorisiert ist. Da allerdings die Autorisierungselemente für die API allesamt in der App und nicht in einem

Browser gespeichert sind, können diese Angriffe auf diese Weise bei Game of TUK nicht funktionieren.

Eine Ausnahme bildet hierbei die Akquisition des Keys während des Anmeldeprozesses, allerdings wird hier die Webseite auch nur zur Anzeige des Keys verwendet. Die Authentifikation erfolgt über Shibboleth, welches gesondert gegen derlei Angriffe gesichert ist.

Zusätzlich existiert noch ein Admin-Interface, über das Models verwaltet werden können. Dieses ist allerdings von Django selber generiert, und besitzt daher die in Django eingebauten Sicherheitsmaßnahmen, wie auch Schutz gegen Cross-Site-Scripting und Cross-Site-Request-Forgery. [19b] Diese sind im Allgemeinen standardmäßig aktiviert, welche somit explizit deaktiviert werden müssen, damit sie nicht genutzt werden. Durch die Nutzung von Models werden außerdem SQL-Injections verhindert, da Django die letztendlichen Anfragen an die Datenbank parametrisiert, womit kein eingefügter SQL-Befehl als solcher interpretiert und ausgeführt wird.

Weiterhin ist auch Phishing keine Angriffsmöglichkeit, denn hierfür müsste der Angreifer einen manipulierten Link an das Opfer leiten, welcher auf den Link klicken und auf der so verlinkten Seite Zugangsdaten eingeben müsste. Allerdings ist die einzige Stelle, an denen ein Spieler überhaupt irgendwelche Daten außerhalb der App eingibt, die Shibboleth-Webseite. Da die Anfrage, die auf Shibboleth weiterleitet, allerdings keinerlei Parameter entgegen nimmt, sondern die Accountzuweisung im Hintergrund auf dem Server erfolgt, gibt es also gar keine Möglichkeit, diese Anfrage zu manipulieren.

Ein Phishing-Angriff könnte somit höchstens auf Shibboleth selber durchgeführt werden, um so an die Zugangsdaten der Nutzer zu gelangen. Allerdings sind diese Zugangsdaten sehr sicherheitskritisch, da diese beispielsweise auch für Anmeldungen zu diversen Vorlesungen in OLAT verwendet werden können. Daher besitzen die Nutzer hier ein anderes Sicherheitsbewusstsein und geben ihre Daten nur in die ihnen bekannte Shibboleth-Login-Webseite ein. Dies kann somit als weiteres Sicherheitsfeature angesehen werden.

Obwohl Denial-of-Service Attacken hingegen prinzipiell immer möglich sind, werden diese im konkreten Fall grundsätzlich vom Host verhindert, da dieser bei zu vielen gleichzeitigen Anfragen den SSL-Handshake verweigert. In Abschnitt 5.4 werden hierbei die Ergebnisse eines Stresstests dargestellt, die auf dem Server ausgeführt wurden.

5.2 Man in the Middle

Um überhaupt einen Man-in-the-Middle Angriff durchführen zu können, muss sich ein potentieller Angreifer zuerst einmal in die Mitte einklinken. Da Game of TUK über HTTPS mit dem Server kommuniziert, müsste die App dem Zertifikat des Angreifers vertrauen. Dazu müsste das Zertifikat auf dem Mo-

biltelefon des Opfers installiert sein. Dies könnte ein Angreifer beispielsweise erreichen, indem er einen mobilen WLAN-Hotspot öffnet, der den Nutzer dazu auffordert, zuerst ein solches Zertifikat zu installieren, da dies aus Sicherheitsgründen für die Nutzung des Hotspots nötig sei. [Eik19] Theoretisch könnte der Angreifer ab diesem Zeitpunkt den gesamten Datenverkehr der App mitlesen.

Allerdings gibt es seit Android 7 eine neue Sicherheitsvorkehrung, nach der selbst-installierten Zertifikaten nicht mehr vertraut wird, beziehungsweise die App dies explizit erlauben muss. [1] Eine ähnliche Sicherheitsvorkehrung gibt es auch bei iOS seit Version 10.3. [2] Daher ist es bei Handys dieser Version oder höher nicht mehr möglich, ohne viel Zutun des Nutzers dessen Datenverkehr abzuhören.

Angenommen der Nutzer hätte nun ein älteres Handy oder würde dem Zertifikat auf sonstige Weise vertrauen. Dann erhält der Angreifer tatsächlich alle übertragenen Anfragen inklusive der zugehörigen Antworten im Klartext und kann diese nach Bedarf manipulieren. Da allerdings wie in Kapitel 4.5.1 beschrieben, der Angreifer das Secret des Client nicht kennt, kann er eine Anfrage so nicht manipulieren oder eine komplett neue Anfrage generieren. Zudem müsste er dafür überhaupt erst wissen wie der Hash zustande kommt, wofür er auf jeden Fall die App dekompilieren müsste. In diesem Fall hat er aber sowieso wesentlich weitreichendere Möglichkeiten; diese Problematik wird in Abschnitt 5.3 erläutert.

Insgesamt könnte der Angreifer also maximal eine getätigte Anfrage wiederholen, was allerdings bei allen Weeklies, mit Ausnahme der Fahrrad-Kilometer zu Fehlern führt. Allerdings hätte letzteres für den Angreifer, außer er greift sich selber an, keinen Nutzen und würde insbesondere auf dem Server auffallen, da der Spieler dieselbe Aktion zur selben Zeit zwei Mal ausführt.

Die einzige Möglichkeit, in der der Angreifer tatsächlich von einem solchen Man-in-the-Middle Angriff profitieren würde, wäre, wenn er die Übergabe des Secrets während des Anmeldeprozesses abhören könnte. Allerdings würde das alleine noch nicht ausreichen, da er, zum Ausnutzen dieses Wissens, trotzdem ebenfalls noch Interna der App benötigen würde. Das bedeutet hierbei explizit, dass er zuerst noch herausfinden müsste, wie dieses Secret im Kontext von folgenden Anfragen genutzt wird.

Da es insgesamt aber sehr unwahrscheinlich ist, dass diese Umstände alle zusammen kommen, um tatsächlich die Anfragen eines anderen Spielers auslesen zu können, und dazu noch der daraus resultierende Profit sehr gering ist, ist dies im Sinne der App-Sicherheit für die Spielrunde 2019 vollkommen ausreichend gewesen. Dennoch ist es eines Ziele dies in der nächsten Spielrunde noch zu verbessern, siehe Kapitel 6.4.

[1] `https://android-developers.googleblog.com/2016/07/changes-to-trusted-certificate.html`

[2] `https://support.apple.com/en-us/HT204477`

5.3 Dekompilierung der App

Da prinzipiell jede Android-App relativ einfach dekompiliert werden kann, ist die einzige Sicherheitsmaßnahme, die hiergegen getroffen werden kann, das Auslesen und Erkennen von Strukturen innerhalb des Dekompilats so schwer wie möglich zu machen. Da allerdings die Client-Apps auf den ursprünglichen Versionen von 2018 basieren, welche aufgrund einiger suboptimal implementierten Bibliotheken nicht mit beispielsweise ProGuard obfuscated werden können, wurde von einer Code-Obfuscation in der neuen Version ebenfalls abgesehen, da eine Umstellung der Bibliotheken zu zeitaufwendig gewesen wäre.

Daher ist der Code der Android-App, wenn sie dekompiliert wird, exakt so lesbar, wie er implementiert wurde. Dies allein reicht aber, im Gegenzug zur Version von 2018 noch nicht, um beliebige Anfragen an den Server zu versenden. So wird, wie bereits in Kapitel 4.5.1 beschrieben, für jede Anfrage das Secret benötigt, das der Client vom Server erhält. Um dieses aus der App auszulesen, müsste der Spieler allerdings sein Handy gerootet haben, um Zugriff auf dieses Secret zu erhalten. Alternativ könnte er durch einen auf sich selbst durchgeführten Man-in-the-Middle Angriff dieses Secret abfangen, oder er lässt sich selbst in einem eigenen Client ein Secret geben. In letzterem Fall wird dadurch aber der echte Client ausgeschlossen.

Frühestens dann könnte der Spieler auf diese Weise selber Anfragen stellen. Dazu müsste er allerdings widerum eine eigene Schnittstelle bauen, die den SHA-256-Hash über Parameterliste und Secret berechnet und mitschickt. Diese Schnittstelle müsste dabei aus Plausibilitätsgründen auch dafür sorgen, dass die Zeitstempel nicht gleich, sondern in realitätsnahen Abständen gesetzt werden, um eine Identifikation des Cheatens auf dem Server zu erschweren. Dies erhöht den Aufwand und stellt somit ein weiteres Hindernis beim Cheaten dar.

Insgesamt ist es also keinesfalls unmöglich, durch Dekompilieren den Server angreifbar zu machen. Allerdings benötigt dies wesentlich mehr Aufwand als in der Spielrunde 2018, was einen Großteil der potenziellen Cheater daher vermutlich bereits von ihrem Vorhaben abhält. Ansonsten können so auch lediglich echte Aktionen simuliert werden, denn sonstige Aktionen oder doppelte Aktionen werden, wie im Kapitel 4.5.2 beschrieben, nicht akzeptiert.

Somit könnte ein Spieler auch definitv nur eine gewisse, maximale Anzahl an Punkten pro Tag erhalten, die offensichtlich durch die maximale Anzahl der legitimen Punkte an diesem Tag beschränkt ist. Um also einen nennenswerten Effekt für das Haus zu erzielen, müsste er auch noch andere Spieler nach ihren Keys fragen, um über deren Account ebenfalls Punkte zu sammeln. Der Aufwand hierfür würde damit aber direkt wieder steigen, zudem wäre hierbei die Gefahr hoch, dass dies nach einiger Zeit auffällt.

Ansonsten wäre das Einzige, was bei diesem Punkt noch verbessert werden kann, die genannte Obfuscation durchzuführen, sodass es nochmal deutlich erschwert wird, das Dekompilat zu verstehen.

5.4 Denial of Service

Mit dem Tool „ab“ von Apache [3] wurde auf dem Server ein Stresstest durchgeführt, um zu überprüfen, wie gut dieser mit einer großen Menge von gleichzeitigen Anfragen umgehen kann. Es gibt hierbei drei Webserver, an die im Zuge eines Tests Anfragen geleitet werden können. Diese sind der Django-Anwedungsserver, ein Apache-Webserver als Proxy auf dem gleichen Gerät und ein nginx-Webserver auf `www.uni-kl.de`. Um Netzwerkeffekte während des Tests auszuschließen, wurde der Test daher nur auf dem Gerät des Django-Anwendungsservers ausgeführt. Somit wurden die Anfragen einmal direkt an den lokal laufenden Spiel-Server gesendet, also an

„127.0.0.1:<port>/gameoftuk/api/<beliebige Funktion>“

sowie einmal mit der URL über den Reverse-Proxy auf dem gleichen Gerät:

„https://www.uni-kl.de/gameoftuk/api/<beliebige Funktion>“

Letzteres entspricht dabei der Art und Weise, wie auch die App mit dem Server kommuniziert, nur dass die Anfragen hierbei natürlich noch weitere Knotenpunkte passieren. Mit der Konfiguration von 1000 Anfragen, bei denen jeweils 100 Anfragen gleichzeitig laufen, ergeben sich die folgenden Ergebnisse:

Dauer in ms	Server	URL
Minimum	53	221
Mittelwert	861	700
Median	131	646
Maximum	27494	1707

Es lässt sich leicht erkennen, dass beim direkten Zugriff auf den Server 95% der Anfragen innerhalb von 5 Sekunden beantwortet werden, wohingegen die restlichen 5% stark verzögert oder sogar innerhalb eines gewissen Zeitlimits gar nicht beantwortet werden. Werden die Anfragen hingegen über den Reverse-Proxy geleitet, so werden alle Anfragen in weniger als 2 Sekunden beantwortet. Dies liegt vermutlich daran, dass diese Anfragen über den Proxy getaktet an den Server weitergeleitet werden, welcher daher nicht mehr so viele, tatsächlich gleichzeitige Anfragen erhält.

Erhält der Reverse-Proxy hingegen deutlich mehr als 100 gleichzeitige Anfragen, so verweigert dieser einigen dieser Anfragen den SSL-Handshake, womit diese also abgelehnt werden. Hieraus lässt sich also schließen, dass eine DoS-Attacke keinen Erfolg hätte, da eine Vielzahl dieser Anfragen von dem Proxy abgelehnt werden würden.

Bezieht man dies nun auf den realen Anwendungsfall, anstatt auf eine gezielte Attacke, so stellt man fest, dass es bei Game of TUK niemals so viele Mitspieler gibt, dass eine solche Anzahl an Anfragen wirklich gleichzeitig an den Server gestellt werden. Somit kann es also auch nicht passieren, dass der Server un-

[3] `https://httpd.apache.org/docs/2.4/programs/ab.html`

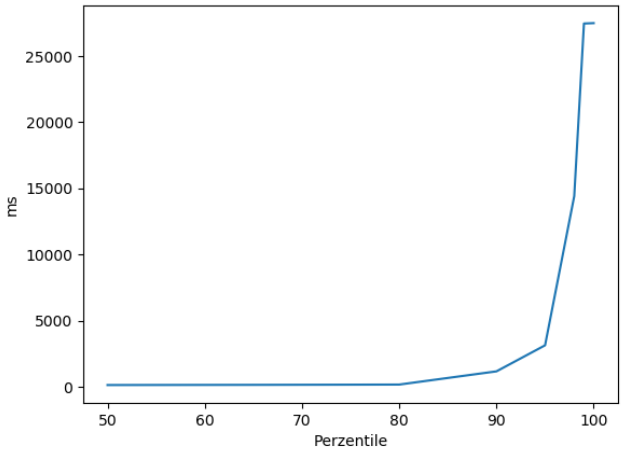

Abbildung 5.1: *Server*

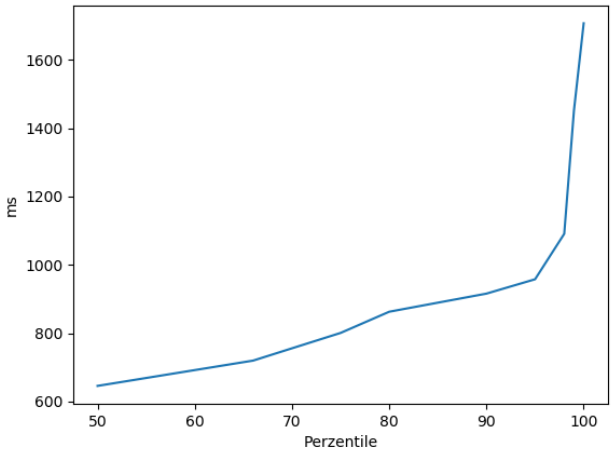

Abbildung 5.2: *URL*

begründet Anfragen ablehnt. Als positiven Nebeneffekt haben wir hiermit also auch gleichzeitig einen Lastentest durchgeführt und die Alltagstauglichkeit des Servers unter Beweis gestellt.

6 Umsetzung Spielrunde 2019 und Future Work

An dieser Stelle soll darauf eingegangen werden, inwieweit die genannten Features und Fehlerbehebungen tatsächlich umgesetzt wurden. Im Anschluss wird dann erörtert, welche neue Möglichkeiten der Absicherung für künftige Spielrunden erschlossen wurden. Dazu muss zuerst gesagt werden, dass die Restrukturierung als solche letztendlich tatsächlich erfolgreich war. So sind sowohl Android als auch iOS an den neuen Server gekoppelt, über welchen auch die gesamte Spielrunde 2019 durchgeführt wurde. Zudem wurden auch speziell die Authentifizierungs- und Validierungs-Maßnahmen, genau so wie diese zu Beginn spezifiziert wurden, komplett implementiert. Allerdings gab es dennoch während des gesamten Entwicklungsprozesses einige Probleme. Diese haben insbesondere dazu geführt, dass der Spielzeitraum um einige Wochen nach hinten verschoben werden musste. Unabhängig davon war zudem der Rollout nicht ganz sauber, wodurch einige Fehler noch während der Spielrunde behoben werden musssten. Dies hatte vor allem starke Auswirkung auf die iOS-Version der App, da diese von einem externen Entwickler programmiert wurden, was den Kommunikationsprozess erschwerte.

6.1 Entwurfsphase

In der Entwurfsphase wurde insbesondere das Lastenheft zur Spezifikation der API angefertigt. Dieses war allerdings zu Beginn der Implementierungsphase immer noch nicht vollständig fertig. Dies führte auch dazu, dass daraus niemals ein Pflichtenheft konkretisiert wurde. Der kritischste Punkt hierbei war, dass zwar die Spezifikationen aller Schnittstellen vorhanden waren, allerdings keinerlei Übersicht darüber, wie diese eigentlich zusammenhängen, beziehungsweise was eigentlich alles geändert werden soll. Dies führte dazu, dass der externe iOS-Entwickler, der nicht an den wöchentlichen Besprechungsrunden beteiligt war, bis zum Beginn der Implementierungsphase nicht wusste, was er genau zu tun hatte.

6.2 Implementierungsphase

Weiterhin war die Koordination zwischen allen Beteiligten nicht optimal, daher wurden stellenweise während der Entwicklung an Server, Android-Client und

iOS-Client an drei verschiedenen Stellen gearbeitet. Da zum Testen der Client-Funktionalitäten die jeweiligen Server-Elemente funktionsfähig sein müssen, konnten so einige dieser Funktionalitäten erst wesentlich später getestet werden als nötig gewesen wäre. Da nach Hochrechnung des restlichen Aufwands festgestellt wurde, dass der ursprünglich angesetzte Termin für die Spielrunde nicht mehr haltbar war, wurde entschieden, diesen um drei Wochen nach hinten zu verschieben. Eine weitere Verschiebung war dabei nicht möglich, da ansonsten der Spielzeitraum mit der Prüfungsphase kollidiert wäre, was drastische Auswirkungen auf die Teilnehmerzahl gehabt hätte.

Zu diesem Termin waren dann auch soweit alle wichtigen Komponenten fertig – der Server und der Android-Client waren sogar schon eine Woche vorher online und konnten getestet werden –, allerdings wurde die iOS-Version aufgrund eines Kommunikationsproblems erst an dem Tag, an dem die App eigentlich im App Store sein sollte, in den Review-Prozess von Apple gegeben, wodurch sich dessen Release um weitere zwei Tage verzögerte.

6.3 Spielphase

Da hierdurch die Spieler der iOS-Version einen starken Nachteil hatten, wurde schnell entschieden, dass diese später in der Wertung gesondert betrachtet werden. Zum gleichen Zeitpunkt wurde aber auch ein Fehler in der Android-Version festgestellt, durch den sich einige Spieler nicht registrieren konnten. Dies war das Ergebnis eines Fehlers im Algorithmus zur Berechnung von SHA-256, der allerdings nur in einigen Sonderfällen auftritt, weshalb dieser Fehler beim Testen nicht aufgefallen ist. So wurde bei dem Algorithmus mit Hex-Zahlen gearbeitet die am Ende zu einem String konvertiert wurden. Allerdings wurden durch die Repräsentation als Zahl, führende Nullen im konvertierten String nicht übernommen, wodurch dieser am Ende zu kurz war.

Da außerdem die erste Android-Version, die im Play Store herunterladbar war, noch nicht SHA-256 zum hashen benutzt hat, da dies während der Testphase eine genaue Fehlersuche erschwert hätte, musste der Server ebenfalls angepasst werden, um die Spieler dieser ersten Version nicht auszusperren. Technisch gesehen ist damit allerdings die sichere Authentifikation verloren gegangen, daher wurde dies nach einer gewissen Zeit, die allen Spielern das Updaten ermöglichte, wieder geändert. Zu diesem wurde allerdings auch festgestellt, dass es versäumt wurde, die geplante Versionskontrolle in die App zu integrieren.

Das Feature, wodurch Spieler ein Duell bei den Quickies wechseln oder zurückstellen können, wurde ebenfalls erst während der Spielrunde – und dort nur auf Android – ermöglicht. Dies lag daran dass die an der Universität beteiligten Entwicklern keinen Zugriff auf den Code für die iOS-App hatten und der externe Entwickler zu diesem Zeitpunkt nicht mehr an der App gearbeitet hat.

6.4 Future Work

Wie bereits in einigen Kapiteln erwähnt, gibt es an einigen Stellen noch Verbesserungsbedarf. So sollte die Übertragung des Secrets vor Man-in-the-Middle-Angriffen auf Client-Geräte mit kompromittiertem Zertifikatsspeicher geschützt werden. Dies könnte beispielsweise explizit durch ein asymmetrisches Verschlüsselungsverfahren wie RSA realisiert werden. Allerdings wäre es vermutlich sinnvoller, Certificate-Pinning zu nutzen, da hierbei die gesamte Kommunikation über TLS/SSL verschlüsselt wäre. Dies hätte dann den Vorteil, dass Man-in-the-Middle Angriffe selbst bei einem kompromittierten Zertifikatsspeicher des Client-Betriebssystems im Allgemeinen nicht mehr möglich wären.

Durch Umfragen wurde zudem festgestellt, dass viele Spieler nicht zufrieden mit dem Fahrrad-Kilometer Weekly waren, da bei diesem zu einfach gecheatet werden kann, beispielsweise indem ein Spieler währenddessen mit dem Bus fährt. Dieser fährt nämlich im Durchschnitt eine Geschwindigkeit die genau in dem zulässigen Intervall liegt. Dabei ist es für den Server natürlich schwer zu überprüfen, wie derjenige Spieler sich fortbewegt hat. Daher sollte dieses Weekly in einer zukünftigen Version definitiv durch ein neues ersetzt werden.

Obwohl durch die neue Architektur wesentlich weniger Fehler oder Möglichkeiten zu cheaten vorhanden waren, ist leider dennoch die Spieleranzahl stark zurückgegangen. Dies liegt vermutlich daran, dass die App in ihren Grundzügen gleich geblieben ist und wenig neuen Inhalt zu bieten hatte. Auf der anderen Seite ist der hauptsächlich verbesserte Aspekt, die Sicherheit, nicht direkt ersichtlich. Dieser sollte daher stärker kommuniziert werden. Das gilt insbesondere, da in der Spielrunde 2018 an einigen Stellen einige Spieler sehr offensichtlich geschummelt haben, daher ist es wichtig anzukündigen, dass dies nun nicht mehr so einfach möglich ist. So können eventuell wieder mehr Spieler das Vertrauen in das Spiel erlangen.

Unabhängig davon wäre es sinnvoll, über eine Umstrukturierung des eigentlichen Spielkonzepts nachzudenken. Dadurch neu entstehende Features müssten dementsprechend von Grund auf neu entwickelt werden.

Im Zuge dessen wurde während einer Nachbesprechung zu Game of TUK auch überlegt, die Clients neu aufzubauen, und dabei ein Framework zu benutzen, das sowohl auf Android als auch auf iOS lauffähig ist. Dies würde den Entwicklungsaufwand massiv reduzieren und zudem Koordinierung vereinfachen, da die Basis beider Clients die Gleiche wäre. So müssten insbesondere Sicherheitsdetails nur noch gegen einen Client getestet werden, sowie nicht mehr so stark auf Eigenheiten der verschiedenen Betriebssysteme geachtet werden. Dies würde schlussendlich auch wiederum die Komplexität des Servers senken.

6.5 Praktische Erfahrungen

Im Zuge der Entwicklung dieses Systems habe ich einige Sachen gelernt. So ist insbesondere Kommunikation während eines solchen Projekts eines der wichtigsten Dinge, da es ansonsten sehr schnell zu Problemen kommen kann, wenn nicht jedem klar ist, was er zu tun hat. An der selben Stelle ist auch die Struktur des Teams eine wichtige Komponente, da die Zuständigkeiten klar geregelt sein sollten. Dies führt ansonsten dazu, dass Anfragen die an die falsche Person gestellt werden, erst nach wesentlich längerer Zeit und über Umwege den eigentlichen Ansprechpartner erreichen.

Auf der technischen Seite habe ich mich insbesondere mit dem Django-Framework auseinandergesetzt. Hierbei habe ich festgestellt, dass dieses zum API-Bau ideal geeignet ist, da es dem Entwickler viel Arbeit in mehrerlei Hinsicht abnimmt. So ist beispielsweise die Schnittstelle zur Datenbank nur an einer Stelle im Code anzugeben, der Rest der Kommunikation wird von dem Backend des Frameworks übernommen. An dieser Stelle wird auch schnell klar, dass durch die Benutzung des MVT-Musters auch die Zuständigkeiten innerhalb des Codes eindeutig identifizierbar sind. Die dadurch gegebene Struktur der API macht daher die Entwicklung sowie die Wartung sehr einfach.

Was die Sicherheit belangt, so wird durch die Nutzung von Django dem Entwickler ebenfalls einiges an Arbeit abgenommen. So muss er sich nicht um einige der kritischsten sicherheitsrelevanten Aspekte kümmern, da dies automatisch passiert. Im Zuge dessen habe ich mich aber auch mit ebendiesen Aspekten beschäftigt, um ihre Relevanz bezüglich Game of TUK beurteilen zu können. Dabei sind auch einige Angriffsmöglichkeiten klar geworden, die wir zuvor während der Entwicklung nicht direkt bedacht haben. So wurden vorhandene Sicherheitsmaßnahmen wie der durchgehende Einsatz von HTTPS noch nicht durch Maßnahmen wie Certificate Pinning verbessert, um Man-in-the-Middle Angriffe gänzlich zu verhindern.

In jedem Fall ist die neue Architektur nun so aufgebaut, dass Cheating nicht mehr trivial und mit mehr Aufwand verbunden ist. Die Haupterkenntnis ist hierbei, dass es nun ein Limit an Punkten gibt, das ein Spieler niemals übersteigen kann, da dies von der neuen Server-Komponente überprüft und verhindert wird. Insbesondere dürfte alleine die Authentifizierung einen Spieler hiervor abschrecken, da er dies nicht mehr anonym machen kann, sondern zumindest seinen Nutzernamen nennen muss. Die Nutzung von Shibboleth ist hierbei ideal, denn dieses kann nahtlos in das System der Dienste der Uni eingegliedert werden. Außerdem können hierdurch mehrere Aspekte der Authentifikation gleichzeitig geregelt werden. So kann einerseits die Zugehörigkeit des Spielers zur Uni überprüft werden, andererseits ein Spiel-Account erstellt werden, ohne dass echte Nutzerdaten verarbeitet und gespeichert werden müssen.

Schlussendlich habe ich also im Gesamten viel über Websicherheit im Allgemeinen, Anti-Cheat-Maßnahmen für Spiele, sowie Entwicklung eines Software-Projektes im Ganzen, inklusive jedweder Probleme gelernt.

7 Fazit

Einer der Haupt-Kritikpunkte von Game of TUK 2018 war dessen Sicherheit, da sie Angriffen aller Art Tür und Tor offen legte. Hierbei war insbesondere Cheating ein großes Problem, was viele Spieler demotivierte, weiterhin an dem Spiel teilzunehmen. Um dieses Problem zu lösen, wurde also unter anderem eine neue Server-Komponente entwickelt und implementiert.

Dazu wurden zu Beginn die Probleme analysiert, die zu der Unsicherheit des Spiels der Spielrunde 2018 geführt haben. Zu diesen wurden dann Maßnahmen entwickelt und implementiert, soweit dies unter den gegebenen Zeit- und Ressourcen-Limitationen möglich war. Hierbei wurde ebenfalls der Hauptaugenmerk auf die Cheating-Problematik gelegt, da diese den Spielern während der Spielrunde am stärksten auffällt und das Vertrauen in die App schädigt. Das Ergebnis ist eine Architektur, die es nur unter erhöhtem Aufwand erlaubt, derlei Angriffe durchzuführen. Gleichzeitig wurde hierbei aber auch der Nutzen gesenkt, da diese Angriffe nur noch einen beschränkten Nutzen haben. Dies liegt daran dass der Server jede Anfrage zuvor validiert und auf Plausibilität prüft. Insgesamt wurde also ein System geschaffen, dessen Aufwand-Nutzen-Verhältnis in der Runde 2019 wesentlich schlechter für einen Angreifer ist, als in der Runde 2018.

Am Ende wurden dann noch einige Angriffs-Szenarien durchgesprochen. Hierbei hat sich herausgestellt, dass das neue System dagegen ebenfalls relativ sicher ist, jedoch besteht dort noch am meisten Verbesserungspotential in der Zukunft.

Zusammenfassend lässt sich also sagen, dass das Ziel der Verbesserung der Sicherheit erreicht wurde, wodurch Cheating in der Spielrunde 2019 so gut wie eliminiert werden konnte.

Literatur

[18] *GAME OF TUK*. Online verfügbar unter https://www.uni-kl.de/ueber-die-tuk/leben-und-kultur/game-of-tuk/. 2018.

[19a] *Django FAQ*. Webseite. Online verfügbar unter https://docs.djangoproject.com/en/2.2/faq/general/#django-appears-to-be-a-mvc-framework-but-you-call-the-controller-the-view-and-the-view-the-template-how-come-you-don-t-use-the-standard-names. 2019.

[19b] *Security in Django*. Webseite. Online verfügbar unter https://docs.djangoproject.com/en/2.2/topics/security. 2019.

[Cho14] Ming Chow. *Abusing Mobile Games*. Präsentation. Online verfügbar unter https://mchow01.github.io/docs/AbusingMobileGames.pdf. 2014.

[Eik19] Ronald Eikenberg. „Sicher unterwegs? Gefahren für Technik auf Reisen“. In: *c't 14/2019* (2019).

[Fos13] Karl Fosaaen. *Hacking iOS Game Center and Passbook with Proxies*. Louisville Metro Infosec Conference Presentation. Online verfügbar unter http://louisvilleinfosec.com/wp-content/uploads/2013/02/KarlFosaaen-iOS-GC-and-PB.pdf. Okt. 2013.

[FT00] Roy T. Fielding und Richard N. Taylor. *Architectural styles and the design of network-based software architectures*. Bd. 7. University of California, Irvine Doctoral dissertation, 2000.

[Grü+18] J. Grützmacher, B. Gusy, T. Lesener, S. Sudheimer und J. Willige. *Gesundheit Studierender in Deutschland 2017*. Studie. Online verfügbar unter https://www.ewi-psy.fu-berlin.de/einrichtungen/arbeitsbereiche/ppg/bwb-2017/_inhaltselemente/faktenblaetter/Gesamtbericht-Gesundheit-Studierender-in-Deutschland-2017.pdf. Ein Kooperationsprojekt zwischen dem Deutschen Zentrum für Hochschul- und Wissenschaftsforschung, der Freien Universität Berlin und der Techniker Krankenkasse, 2018.

[Les+18] T. Lesener, W. Blaszcyk, B. Gusy und M. Sprenger. *Wie gesund sind Studierende der Technischen Universität Kaiserslautern?* Studie. Online verfügbar unter https://www.uni-kl.de/fileadmin/sgm/PDF/UHR_2018_TU_Kaiserslautern.pdf. Technische Universität Kaiserslautern und Freie Universität Berlin, 2018.

[Ree79] Trygve Mikkjel Heyerdahl Reenskaug. „The original MVC reports“. 1979.

[YC02] Jianxin Jeff Yan und Hyun-Jin Choi. „Security issues in online games“. In: *The Electronic Library* 20.2 (2002), S. 125–133.

[YPK13] Amir Yahyavi, Jeffrey Pang und Bettina Kemme. „Towards providing security for mobile games“. In: *Proceedings of the Annual International Conference on Mobile Computing and Networking, MOBICOM*. Okt. 2013, S. 47–52. DOI: 10.1145/2505906.2505912.