Inhaltsverzeichnis

Abkürzungsverzeichnis

API ..... *Anwendungsprogrammierschnittstelle*
BSI ..... *Bundesamt für Sicherheit in der Informationstechnik*
CLOUD ..... *Clarifying Lawful OVerseas Use of Data*
CSP ..... *Cloud Service Provider*
DoS ..... Denial-of-Service
IaaS ..... *Infrastructure-as-a-Service*
IP ..... Identity Provider
JWT ..... *JSON-Web-Token*
PaaS ..... *Platform-as-a-Service*
SaaS ..... *Software-as-a-Service*
SPA ..... *Single Page Applikation*

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Motivation

Gestützt durch Vorfälle aus jüngster Zeit ist die Sicherheit in der Informationstechnologie für Firmen auf der ganzen Welt zu einem immer mehr an Bedeutung gewinnenden Bestandteil geworden. Als Folge der aktuellen Angriffe wie zum Beispiel das unautorisierte Entwenden von mehr als 500 Millionen Nutzerdaten von Facebook [1] oder das Ausnutzen der Schwachstellen der Microsoft Exchange Server in zahlreichen Unternehmen [2] fokussieren immer mehr Firmen die IT-Sicherheit ihrer Produkte. Dies gilt nicht nur für Software wie native Applikationen auf Mobiltelefonen oder Desktopanwendungen, sondern auch für Anwendungen, die ihren Nutzern als Clouddienste zur Verfügung stehen und öffentlich über das Internet erreichbar sind. Jedoch sind Kriterien, die für die Absicherung dieser Dienste gedacht sind, wie zum Beispiel des National Institut for Standards and Technology (NIST), der Cloud Security Alliance (CSA) oder das Bundesamt für Sicherheit in der Informationstechnik (BSI) nur wenig verbreitet und werden von den betreffenden Firmen partiell umgesetzt. Des Weiteren ist festzustellen, dass sich die aufgezählten Kriterienkataloge immer wieder aufeinander beziehen, jedoch keine einheitliche Norm zur Regelung und Umsetzung der Sicherheitskriterien vorliegt. Als Richtlinie und Maßstab für zukünftige Entwicklungen an Cloudprodukten sollte jedes Unternehmen einen Katalog an bestehenden Sicherheitsmaßnahmen und Regularien entwerfen, der den Endnutzern ein gewisses Maß an Sicherheit garantiert.

1.2 Unternehmensvorstellung

Die STP Informationstechnologie GmbH (nachfolgend kurz: STP GmbH) ist ein in Karlsruhe gegründetes IT-Unternehmen. Der Schwerpunkt der angebotenen Softwarelösungen und Informationssysteme zielt auf die Anwendungen in den Berufsgruppen im Rechtssektor wie Anwälte, Justizverwaltungen und weiteren fachnahen Institutionen ab. Gegründet wurde das Unternehmen im Jahr 1993 von Gunter Thies und Ralph Suikat als „Suikat-Thies + Partner GmbH“. Im Jahre 2001 wurde die Unternehmensform in eine Aktiengesellschaft umgewandelt [3]. Die STP AG im Rahmen eines Zertifizierungsprogramms durch SGS-International Certification Services GmbH nach DIN ISO 9001 zertifiziert. Seit November 2011 gilt dieses Zertifikat für die komplette STP Gruppe: STP Informationstechnologie AG, STP Holding GmbH, STP Portal GmbH und STP Solution GmbH. Seit dem 05. März 2021 ist die STP von einer Aktiengesellschaft in eine GmbH umfirmiert [3].

Intern untergliedert sich die STP weiterhin in einzelne Abteilungen, Arbeitsbereiche und -gruppen, die mit verschiedenen Themen betraut sind. Der Schwerpunkt jeder einzelnen Abteilung liegt auf einem anderen Gebiet der Software- bzw. Produktentwicklung.

Der Fokus bei der Softwareentwicklung liegt jedoch primär auf der Umsetzung von Kundenlösungen mit dem .NET-Framework. Die Produktpalette umfasst neben On-Premise Lösungen, die lokal beim Kunden eingesetzt werden, auch Dienstleistungen wie Consulting- bzw. Beratungslösungen, die von internen Beratern bzw. Fachbearbeitern angeboten werden.

Der Wirkungsbereich der STP Informationstechnologie umfasst die gesamte DACH-Region. Dies bedeutet, es werden neben Kunden aus Deutschland auch Kunden aus der Schweiz, Standort der jüngsten Zweigstelle, und Österreich betreut. Die Niederlassung in Bulgarien fungiert in diesem Unternehmensverbund bisher als Zuarbeiter für spezielle Aufgaben der Entwicklung.

Das hier vorliegende Projekt wurde hauptsächlich in der Abteilung Produktentwicklung (PDE) mit Betreuung durch Manuel Naujoks durchgeführt und erarbeitet. Der Fokus dieser Abteilung liegt auf der Evaluation und Verwendung von neuen Technologien im Ökosystem .NET, die bei der Entwicklung von neuen hauseigenen Produkten verwendet werden sollen.

1.3 Grundlagen

1.3.1 Sicherheitsrelevante Begrifflichkeiten und Verfahren

1.3.1.1 Authentifizierung und Autorisierung

Aufgrund der Relevanz der Begriffe **Authentifizierung** und **Autorisierung** im Bereich der Informationssicherheit werden diese zunächst zum allgemeinen Verständnis in den Kontext eingeordnet und erläutert.

Die Definition des Begriffs der **Authentifizierung** wird anhand des nachfolgenden theoretischen Beispiels von einer Kommunikation zwischen einem menschlichen Nutzer mit einer Anwendung (Maschine) exemplarisch dargestellt.

Die Authentifizierung des Nutzers bei einer Anwendung ist in vielen Fällen der erste Schritt, wenn der Nutzer Zugriff auf geschützte Ressourcen, wie zum Beispiel die Daten seines Profils in einem sozialen Netzwerk haben möchte. Hierzu wird er vom System bzw. der Anwendung zuerst aufgefordert den Usernamen und sein Passwort (nachfolgend Zugangsdaten) einzugeben. Diesen Vorgang der Authentifizierung nutzt die Anwendung, um zu prüfen, ob es sich bei dem vorliegenden Nutzer wirklich um den Nutzer handelt, der er vorgibt zu sein. Stimmen die Zugangsdaten, die meistens via Abgleich von verschlüsselten Werten in der Datenbank überprüft werden, mit den angegebenen Werten überein, ist der Nutzer erfolgreich authentifiziert. Ist dies nicht der Fall, wird dem Nutzer der Zugriff verwehrt. Bei erfolgreicher Authentifizierung erhält der Nutzer in den meisten Fällen ein Token, der ihm als Beweis seiner Identität dient, um sich gegenüber dem System auszuweisen. Somit wird verhindert, dass bei erneuter Interaktion des Nutzers mit dem System, dieser sich erneut anmelden muss.

Bei der **Autorisierung** kommt der bereits genannte Token zum Einsatz. Möchte der Nutzer nun auf die Daten seines Profils zugreifen, sendet er neben dem Request noch seinen Token, im Header des Requests, mit. Dieser Token wird anschließend evaluiert und auf seine Gültigkeit geprüft. Stimmen die notwendigen Rollen mit denen im mitgelieferten Token überein, erhält der Nutzer den gewünschten Zugriff. Ansonsten wird ihm der Zugriff verwehrt.

Die Form und der Informationsgehalt dieser Tokens kann sich in den unterschiedlichen Authentifizierungs-Verfahren (nachfolgend auch Flows genannt) unterscheiden. In den meisten Fällen handelt es sich jedoch um einen sogenannten JSON-Web-Token (JWT), der im Bearer Schema vorliegt. Diese Art von Tokens wird zur Übermittlung von Informationen genutzt und sind in den meisten Fällen signiert und verschlüsselt.

1.3.1.2 JSON-Web-Token

Der grundlegende Aufbau des Tokens lässt sich in drei Bestandteile aufgliedern. Beginnend mit dem „Header“ oder auch Kopf, der den Typ des Tokens und den Algorithmus zur Signatur festlegt. Anschließend folgt durch einen Punkt getrennt der „Payload“, der userspezifische Informationen und die Berechtigungen des Users enthält. Als letzter Bestandteil folgt erneut durch einen Punkt getrennt die „Signature“ oder auch Signatur, mit der die Validität des Tokens überprüft werden kann.

Die gängigste Variante ist das Signieren der Tokens von Seiten des Identity Providers und eine Bereitstellung des Public Keys über einen öffentlichen Endpunkt. Mit Hilfe dieses Public Keys kann anschließend eine Client-Anwendung die Echtheit und die Validität des gesendeten Tokens überprüfen. Zugriff auf diese Informationen erhalten die Clients bzw. Anwendungen über spezielle Endpunkte, die die Identity Provider hierfür zur Verfügung stellen. Der derzeitige Standard zur Ausgabe dieser Informationen definiert eine URL in dem Format `<IdentityProviderDomain>/.well-known/openid-configuration`.

Das nachfolgende Beispiel zeigt einen JWT in seinem kodierten und dekodierten Zustand.

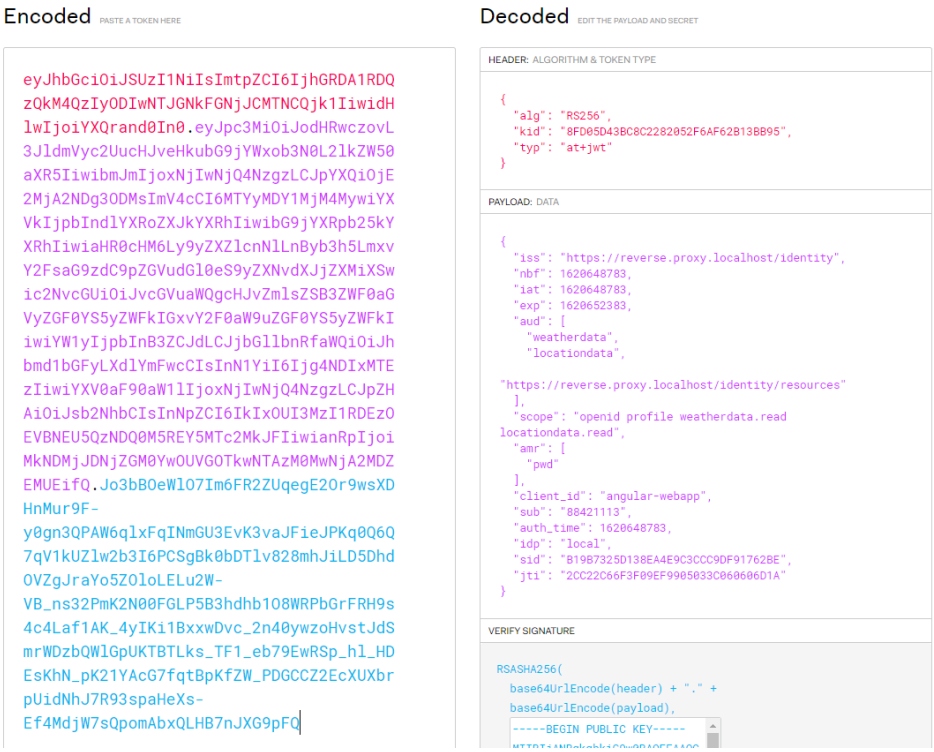

Abbildung 1 JSON-Web-Token in kodierten und dekodierten Zustand
(Quelle: Eigene Abbildung)

1.3.1.3 Authorization Code Flow with Proof-Key-of-Code-Exchange

Zuerst erfolgt einer Erläuterung des "**Authorization Code Flow + Proof Key of Code Exchange (PKCE)**". Diese Authentifizierungsmethode ist derzeit der Standard zur Authentifizierung mittels Webapplikationen, wie einer SPA bei Identity Providern. Der Vorgang besteht darin, dass der Client (hier eine Webapplikation) eine **Code_Challenge** an den Authorization Server (hier den Identity Provider, kurz AS genannt) schickt. Bei dieser **Code_Challenge** handelt es sich um eine Zeichenkette aus Base64-kodierten Werten. Diese Werte können Zeichen im Format von `a-z, A-Z, 0-9, . , ,,‘, ~` und - enthalten.

Anschließend wird diese Zeichenkette mittels SHA-256 verschlüsselt und an den AS geschickt. Nach Eingang der **Code_Challenge** speichert der AS den Wert ab und sendet eine **Code_Verification** zurück.

Möchte der Client nun einen Token zum Zugriff auf geschützte Bereiche des Ressource Server (kurz RS genannt), muss er neben der **Code_Challenge** zusätzlich die **Code_Verification** an den AS schicken, um sich zu authentifizieren [4]. Der Nachteil bei dieser Authentifizierungsvariante besteht darin, dass bei Entwendung des Tokens, dieser für die restliche Zeit seiner Gültigkeit weiterverwendet werden kann, ohne die Möglichkeit des Nutzers, dies zu verhindern.

1.3.2 Angriffsvektoren für Cloud-Applikationen

Zur besseren Einordnung von gängigen Angriffsvektoren auf cloudbasierte Softwarelösungen bzw. alle Anwendungen, die öffentlich über das Internet erreichbar sind, werden nachfolgend die im Blickpunkt dieser Thesis beleuchteten Attacken und die dazu notwendigen Vorkehrungen erläutert.

1.3.2.1 Denial-of-Service Angriff (DoS)

Der Denial-of-Service Angriff oder auch DoS Angriff beschreibt das Überlasten von Webanwendungen mit einer Flut von Anfragen. Ziel des Angriffs ist das vorübergehende oder gänzliche Blockieren eines Dienstes. Hieraus resultiert zum Beispiel eine längere Ladezeit für die Kunden bis hin zur Nicht-Erreichbarkeit. Diese Art von Attacken können auf unterschiedlichen Ebenen des OSI Modells ansetzen. Angriffe auf Ebene drei und vier des Netzwerkmodells haben gezielt das Überlasten von Firewalls und Load Balancer zum Ziel. Angriffe auf Ebene sieben, dem Application Layer, sind auf das Überlasten der Applikation an sich spezifiziert [5].

1.3.2.2 Bot-Netze

Bei einem Bot-Netz handelt es sich um eine Gruppe aus Computern oder Servern, die mit Schadcode infiziert und von einem zentralen Server gesteuert werden. Diese Computer bzw.

Server werden auch als Bots bezeichnet und können innerhalb des Netzwerkes miteinander kommunizieren. Der Schadcode wird üblicherweise über Viren, Trojaner oder Würmer auf die Rechner gebracht und wird dort im Hintergrund ausgeführt. Bot-Netze bestehen zumeist aus mehreren tausend Rechnern und werden für die Ausübung von zum Beispiel Distributed-Denial-of-Service Angriffen verwendet [6, S. 75-76].

1.3.2.3 Distributed-Denial-of-Service Angriff

Bei einer Distributed-Denial-of-Service Attacke handelt es sich um die erweiterte Form des Denial-of-Service Angriffs. Hierbei greift der Angreifer auf die Funktionalität eines Bot-Netzes zurück, das er für das Senden der Anfragen auf das gewünschte Opfer instrumentalisiert. Mithilfe dieser Form des DoS-Angriffs umgeht der Angreifer die begrenzte Bandbreite einer Maschine und nutzt stattdessen die Rechenkapazität von mehreren verteilten Rechnern. Zusätzlich wird durch das Nutzen verschiedener Rechner eine größere Spanne an verwendeten IP-Adressen benutzt. Dies macht die Identifikation von normalem und schädlichem Traffic schwieriger [6, S. 121-123].

1.3.3 Einführung in die Cloud Taxonomie

Für die Einordung und das Verständnis der Cloud Taxonomie werden in diesem Kapitel der Thesis alle notwendigen Begriffe und deren Definition eingeführt. Die Ressourcen eines Cloud Computing Dienstes reichen von Software-Diensten bis zu Datenspeichern, Betriebssystemen und ganzen Hardware-Infrastrukturen. Basierend auf dem Service-Modells des Dienstes kann in Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) und Software-as-a-Service (SaaS) unterschieden werden [7, S. 5]. Neben diesen bereits aufgeführten Service Modellen gibt es noch eine Bandbreite an weiteren Diensten, die über die Cloud angeboten werden können. Dazu zählen zum Beispiel Communication-as-a-Service (CaaS), Compute-as-a-Service (CompaaS), Data-Storage-as-a-Service (DSaaS) bzw. Database-as-a-Service (DaaS) oder Network-as-a-Service (NaaS) [8, S. 17]. In diesem Grundlagenkapitel bzw. im Rahmen der Thesis werden jedoch nur die ersteren drei Modelle genauer beschrieben. Sie sind im Rahmen der Cloud Sicherheit am besten für die Einordnung der Verantwortung des Kunden bezüglich der Wartung und Absicherung der in Anspruch genommenen Cloud Leistung geeignet.

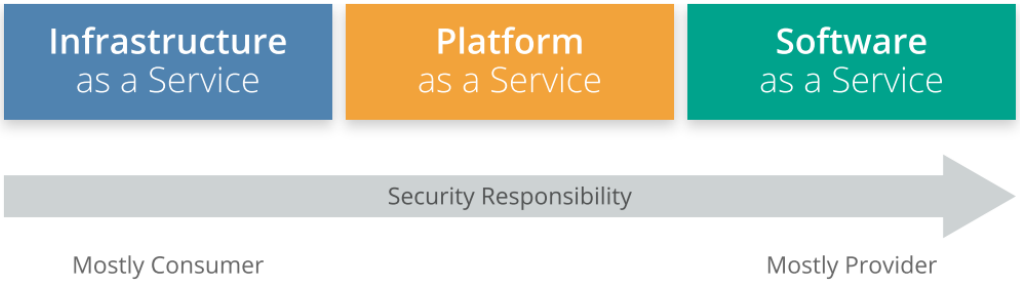

Abbildung 2 Verantwortung über die genutzten Cloud Services.
(Quelle: [9, S. 21])

1.3.3.1 Infrastructure-as-a-Service

Durch die Verwendung von Infrastructure-as-a-Service hat der Cloud Kunde den kompletten Zugriff auf alle Komponenten (wie z.B. Betriebssysteme, Middleware, Laufzeit, Daten und Applikationen) des gemieteten Cloud Servers. Hierauf kann über vordefinierte grafische Benutzerschnittstellen oder VPN Verbindungen zugegriffen werden [8, S. 17]. Dies garantiert neben einer Vielzahl an Konfigurationsmöglichkeiten eine große Verantwortung hinsichtlich der Wartung des Servers und den darauf ausgerollten Applikationen. Zusätzlich muss darauf geachtet werden, dass die neusten kritischen Sicherheits- und Betriebssystem-Updates installiert sind. Dies rundet die Konfiguration der Firewall, die den Server vor unbefugtem Zugriff schützt, ab.

1.3.3.2 Platform-as-a-Service

Im Rahmen von Platform-as-a-Service werden dem Cloud Kunden unterschiedliche Plattformen für das Betreiben seiner Anwendungen zur Verfügung gestellt. In diesem Rahmen kann dieser seine Applikationen mit Hilfe von vorhandenen Entwickler-Tools und Laufzeitumgebungen ohne größeren Aufwand hinsichtlich der Konfiguration platzieren [8, S. 16-17].

1.3.3.3 Software-as-a-Service

Der Cloud-Dienst Software-as-a-Service impliziert eine in der Cloud betriebene Anwendung, auf die der Kunde über einen Web-Browser zugreifen kann. Der Nutzer dieser Software hat somit den Vorteil, dass die Anwendung nicht lokal auf seinem System betrieben werden muss und somit keinerlei lokale Ressourcen verbraucht. Hiermit entfallen der Installationsprozess auf der lokalen Umgebung und der Kauf von Desktop- und Server-Lizenzen. Abgerechnet wird je nach Nutzungsdauer und der Anzahl der Nutzeraccounts, die für den jeweiligen Tenant angelegt wurden. Typische Anwendungen in diesem Service Modell sind E-Mail- und Dokumentenmanagement-Programme [8, S. 16].

Wie in der nachfolgenden Grafik nochmals genauer veranschaulicht, ist innerhalb der einzelnen Service Modelle ein Gefälle an Verantwortung zu erkennen, die der Cloud Kunde selbst übernehmen muss. Von der kompletten Verwaltung von eigenen On-Premise Systemen bis hin zur fremdverwalteten SaaS-Lösung.

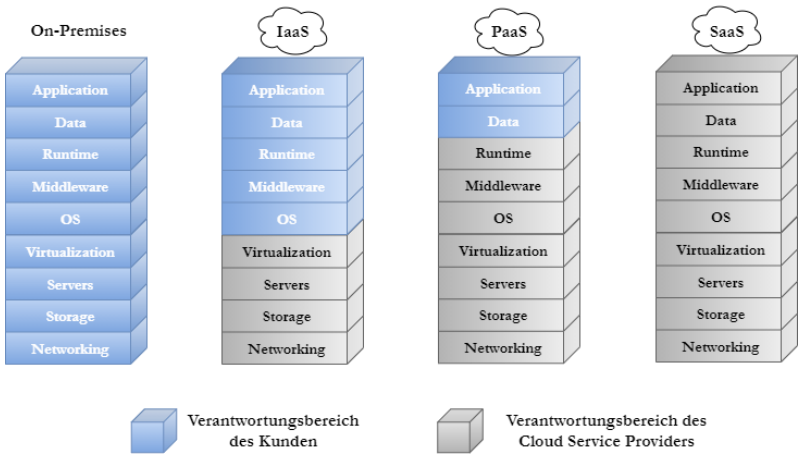

Abbildung 3 Verantwortung von Cloud Kunde und Cloud Betreiber (Quelle: Angelehnt an [8, S. 16])

1.3.3.4 Unterscheidung des verwendeten Deployment-Modells

Im nachfolgenden Kapitel werden die einzelnen Deployment-Modelle, die potentiellen Cloud-Kunden für das Betreiben ihrer Anwendungen zur Verfügung stehen umschrieben und voneinander abgegrenzt.

1.3.3.4.1 Die öffentliche Cloud (Public Cloud)

Im Falle einer "Public Cloud" handelt es sich um eine geteilte Ressource, die dem Endkunden über eine öffentliche sichere IP zur Verfügung steht. Hierbei werden den Endkunden die verwendeten Leistungen des Modells nach dem Pay-as-you-go Prinzip berechnet. Verwendet der Kunde nur wenige Services innerhalb seines eigenen Produktes, fallen die Kosten meistens geringer aus, wie bei einem Kunden mit mehreren Services. Nachteile dieses Modells sind, dass durch gemeinsame Nutzung der virtuellen Maschinen von unterschiedlichen Kunden, keine Garantie auf die Sicherheit und ein Backup der Daten besteht. Hierum muss sich der Kunde selbst kümmern. Die Trennung zwischen den Verantwortungsbereichen der Kunden auf der VM basiert auf einer logischen Trennung, die in Form einer Multi-Tenant-Architektur umgesetzt wird. Typische Beispiele von Public Cloud Modellen sind die frei verfügbaren Speicher für digitale Inhalte von Providern wie Google, Amazon oder Facebook [8, S. 19].

1.3.3.4.2 Die private Cloud (Private Cloud)

Im Rahmen des Private Cloud Deployment Modells handelt es sich um ein Modell, welches innerhalb der eigenen IT-Infrastruktur des betreibenden Unternehmens ausgerollt ist. Hierbei besteht die Möglichkeit die notwendige Infrastruktur In-House zu betreiben oder an einen externen Dienstleister (Dritten) auszulagern und via VPN zuzugreifen. Bei diesem Modell wird im Vergleich zu einer Public Cloud ein höherer Grad an Sicherheit erreicht, da die hierauf betriebenen Anwendungen nicht über ein öffentliches Netzwerk zugänglich sind. Zusätzlich besitzt der Betreiber die Möglichkeit zur Wahl des Standortes für die Speicherung seiner Daten ("data at rest"). Dies ist zum Beispiel bei verteilten Public Cloud Modellen nicht möglich. Zusätzlich besteht die Option einer vorgeschalteten Firewall, die gegen gängige Angriffe aus dem Internet schützen kann [8, S. 19-20].

1.3.3.4.3 Die gemeinschaftliche Cloud (Community Cloud)

Die Eigenschaften einer Community Cloud lassen sich aus den bisher genannten Cloud Modellen der Public und der Private Cloud zusammensetzen. Hinsichtlich des Zugangs ist eine Community Cloud nur Mitgliedern der betreibenden Gemeinschaft zugänglich. Somit ist es wie bei einer Private Cloud nur einem bestimmten Nutzerkreis möglich dieses Modell zu nutzen. Die Ressourcen werden wie bei einer Public Cloud zwischen den Nutzern aufgeteilt und die Trennung der einzelnen Zuständigkeitsbereiche erfolgt nur auf einer logischen Basis. Es besteht hierbei jedoch die Möglichkeit in einem kontrollierten Rahmen Informationen zwischen den unterschiedlichen Mitgliedern auszutauschen. Dies macht dieses Modell für Unternehmen in der Gesundheitsbranche interessant. Das Verwalten der Cloud Infrastruktur kann über die Organisationen erfolgen, die die Dienste nutzen, oder über einen externen Dritten [8, S. 20].

1.3.3.4.4 Die hybride Cloud (Hybrid Cloud)

Eine Hybrid Cloud kann als Konzept für das Verwenden von mehreren einzelnen Cloud Deployment Modellen verstanden werden. Hierbei ist es möglich eine Private mit einer Public oder Community Cloud zu verbinden und die unterschiedlichen Vor- und Nachteile des jeweiligen Modells miteinander zu vereinen. Zum Beispiel wäre es möglich je nach Relevanz der verarbeiteten Daten, Daten mit hohem Sicherheitsniveau in einem privaten Teil der Cloud und Daten mit einem geringeren Niveau an Sicherheit in einem öffentlichen Teil zu speichern [8, S. 20].

1.3.4 Einführung in die ML Taxonomie

In diesem Abschnitt werden die im Rahmen der Thesis verwendeten Konzepte aus dem Bereich des maschinellen Lernens eingeführt und erläutert. Hierbei werden die verwendeten Techniken jedoch nicht mathematisch dargestellt, sondern nur exemplarisch für ihren Einsatzzweck in den Kontext der Arbeit eingeordnet.

1.3.4.1 Entscheidungsbäume

Bei einem Entscheidungsbaum, oder auch Decision Tree, handelt es sich um ein Verfahren des überwachten (supervised) Lernens. Sie werden in den meisten Fällen für Regressionen oder wie im Rahmen dieser Thesis für eine binäre Klassifikation eingesetzt. Da es sich um ein überwachtes Lernverfahren handelt, müssen die Daten entsprechend vorbereitet werden, um den Baum während des Trainings eine klare Vorstellung davon zu geben, welche Kriterien im Ergebnis ausschlaggebend sind. Ein solcher Baum besteht aus mehreren Knoten, Kanten und Blättern. In einem Knoten wird auf Basis der gelernten Daten und deren Attributen somit stets eine Ja-Nein-Frage beantwortet, die je nach Ergebnis in ein bestimmtes Blatt ableitet [10, S. 151-152]. Nachfolgend wird solch ein Baum beispielhaft dargestellt.

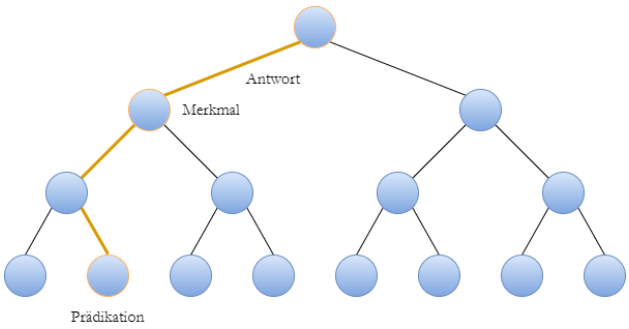

Abbildung 4 Beispielhafte Darstellung eines binären Entscheidungsbaumes (Quelle: Eigene Abbildung)

1.3.4.2 Over-Fitting

1.3.4.3 Random Forest

2 Hauptteil

Inhalt dieses Kapitels ist die theoretische Ausarbeitung des Kriterienkataloges anhand eines vorher vorgegebenen operationalisierbaren Szenarios mit anschließender Konzeption eines Dienstes, der mit passenden Gegenmaßnahmen die negativen Effekte abmildern soll.

2.1 Beschreibung des operationalisierbaren Szenarios

Für die Erarbeitung des Kriterienkatalogs für die STP Cloud und dessen Anspruch an Vergleichbarkeit wird anhand eines der gängigsten Risiken für Anwendungen in der Cloud, eine Situation konstruiert, welche die Basis für die Auswahl der Kriterien für die Gefahrenanalyse bildet.

Die Auswahl des Angriffsvektors basiert auf der Liste der Cloud Security Alliance, die auch in der hier verwendeten Literatur [11] zur Veranschaulichung von Risiken für Cloudanwendungen herangezogen wird:

- Datenpannen
  Ausnutzen von schlecht konfigurierten Datenbanken, über die Angreifer Zugriff auf gespeicherte Kundendaten erhalten können.
- Datenverlust
  Durch Angreifer, unachtsame Service Provider oder Naturkatastrophen ausgelöste Verluste von Daten.
- Übernahme von Account oder Dienst Netzwerkverkehr (Traffic Hijacking)
  Diese Art von Risiko kann unterschiedliche Ursachen besitzen. Ein Angreifer könnte zum Beispiel durch Umleiten von Nutzern auf gefälschte Login-Seiten oder die Manipulation von Daten Zugriff auf Nutzerkonten des Dienstes erlangen und dies als Basis weiterer Angriffsvektoren verwenden.
- Unsichere APIs und Benutzeroberflächen
  Zur Verwaltung der Cloud-Infrastrukturen und deren Überwachung (auch Monitoring genannt) werden von den IT-Administratoren in den meisten Fällen APIs oder grafische Benutzeroberflächen verwendet. Falls diese ohne eine Absicherung frei aus dem öffentlichen Netzwerk adressiert werden können, ist hier eine kritische Sicherheitslücke vorhanden.

- Denial of Service (DoS) Attacken

  Im Rahmen von DoS Attacken können durch das ständige Anfragen der Dienste in der Cloud, Probleme in der Verfügbarkeit auftreten. Dies kann wiederum erhebliche Kosten für den Anbieter der Cloud-Dienste bedeuten, da viele Bezahlmodelle nach dem Pay-As-You-Go Prinzip verwaltet werden. Bedeutet, dass durch die höhere Auslastung der virtuellen Maschinen, diese durch kostspielige Upgrades wie virtuellen Arbeitsspeicher erweitert werden, um die Lastspitzen abfangen zu können.
- Bösartige Insider

  Beschreibt das Risiko eines Angestellten des Cloud-Anbieters oder einer dritten Partei, der durch böswilliges Verhalten dem Cloud Kunden sowie dem Cloud-Anbieter Schaden zufügen kann.
- Missbrauch von Cloudinfrastrukturen (Cloud Abuse)

  Der Missbrauch von Cloudinfrastrukturen stellt das Ausnutzen der Rechenkapazität des verteilten Systems als Risiko dar. Anwendungsfälle wären zum Beispiel die Nutzung der Rechenleistung für das Knacken einer Verschlüsselung oder die Durchführung einer DoS-Attacke.
- Unbedachte Nutzung von Cloud Technologien durch Unternehmen

  Hier wird die unüberlegte Migration von Unternehmenssoftware in die Cloud thematisiert. Durch fehlendes Verständnis der Risiken, die durch eine Cloudmigration entstehen, die für zum Beispiel eine vorherige On-Premise Software nicht relevant waren, müssten sämtliche Kriterien der Cloud Sicherheit bei einer derartigen Entscheidung miteinbezogen werden. In vielen Fällen wird dies wiederrum von Firmen nicht mit ins Kalkül gezogen.
- Sicherheitslücken auf technischer Ebene

  Dieses Risiko macht auf die Gefahren von Sicherheitslücken auf der Ebene der Software, Plattform oder Infrastruktur aufmerksam, auf der die Anwendungen des Cloud Kunden laufen. Hierdurch kann auch eine sichere Anwendung durch Kompromittierung der darunterliegenden Infrastruktur lahmgelegt werden.

Bei der Entscheidung für die Wahl des Angriffsvektors wurden die aktuellen Gegebenheiten der Anwendung berücksichtigt. In Folge des Hostings der Lösung bei einem externen Anbieter und somit der Elimination von einigen der oben genannten Risiken fiel die Entscheidung auf die Denial of Service (DoS) Attacken. Dies lässt im späteren Teil der Implementierung von Gegenmaßnahmen die reine technische Behandlung des Problems, ohne das Einwirken von

dritten, wie zum Beispiel eines menschlichen Nutzers, zu (Social Engineering Attacken müssen somit nicht berücksichtigt werden).

2.2 Bewertungskriterien für die Analyse

Zur Bewertung der SaaS-Lösung wird anhand der Bewertungskriterien des Cloud Computing Compliance Criteria Catalogue des Bundesamtes für Sicherheit in der Informationstechnik C5:2020, der am 21.01.2020 neu aufgelegt wurde, ein Rahmen für das zugrundeliegende Szenario entworfen. Aufgrund seines Umfangs und der Komposition der Kriterien aus dem ISO/IEC 27001 und der Controls Matrix, entwickelt von der Cloud Security Alliance (CSA), stellt der C5 eine fundierte Grundlage für eine Selektion der nachfolgenden Kriterien dar [12, S. 11]. Hierfür wird der Kriterienkatalog zuerst auf die wichtigsten Bewertungskriterien heruntergebrochen. Der komplette Katalog besteht aus 17 unterschiedlichen Bereichen, die wiederum aus 121 Basiskriterien bestehen. Neben der Sicherheit von Softwareanwendungen, die Umsetzung von Sicherheitsstandards in den zu prüfenden Unternehmen werden auch allgemeine Vorkehrungen zur Absicherung bestimmter Bereiche innerhalb von Rechenzentren bewertet. Zu der Bearbeitung der hier vorliegenden Fragestellung ist dieses Sammelwerk zu umfangreich. Für die Bewertung der vorhandenen Sicherheitsmechanismen des Systems bedarf es dem Streichen einzelner Punkte. Zur besseren Übersicht der abgedeckten Bereiche innerhalb des C5 werden diese nachfolgend aufgeführt und mit einer groben Definition ihrer Bewertungsgrundlage beschrieben. Die komplett gestrichenen bzw. gekürzten Bestandteile des Katalogs werden zusätzlich jeweils mit einer Begründung versehen, die den Grund für das Nichtverwenden innerhalb des eigenen Kriterienkatalogs aufzeigt.

2.2.1 Organisation der Informationssicherheit (OIS)

Der Bereich Organisation der Informationssicherheit (OIS) beschäftigt sich mit dem Ziel der "Planung, Umsetzung, Aufrechterhaltung und (der) kontinuierliche(n) Verbesserung eines Rahmenwerks zur Informationssicherheit innerhalb der (zu bewertenden) Organisation" [12, S. 16].

Bezüglich des anzuwendenden Szenarios ergab sich für die sieben Basiskriterien folgendes Ergebnis:

Tabelle 1 Anwendbare Kriterien aus dem Bereich OIS

Anwendbare Kriterien	Nicht-anwendbare Kriterien	
• OIS-05	• OIS-01 • OIS-02 • OIS-03	• OIS-04 • OIS-06 • OIS-07

Die hier formulierten Kriterien bewerten, wie in der Definition bereits angedeutet, die Umsetzung eines Rahmenwerkes, also eines dokumentierten Prozesses innerhalb des Unternehmens, der den Umgang mit Sicherheitsrisiken beschreibt und mögliche Handlungsanweisungen an die betreffenden Personen überträgt. Sie sind somit außer des Kriteriums OIS-05, das den Umgang mit aktuellen Bedrohungen für Dienste in der Cloud beschreibt, nicht für den Einsatz innerhalb des Szenarios zu verwenden.

2.2.2 Sicherheitsrichtlinien und Arbeitsanweisungen (SP)

Im Rahmen der Sicherheitsrichtlinien und Arbeitsanweisungen (SP) steht im Mittelpunkt das "Bereitstellen von Richtlinien und Anweisungen bzgl. des Sicherheitsanspruchs und zur Unterstützung der geschäftlichen Anforderungen" [12, S. 16].

In Anlehnung an die Argumentation aus dem Bereich Organisation der Informationssicherheit (OIS) ist nach der Evaluation der Sicherheitsrichtlinien und Arbeitsanweisungen (SP) folgendes Resultat entstanden:

Tabelle 2 Anwendbare Kriterien aus dem Bereich SP

Anwendbare Kriterien	Nicht-anwendbare Kriterien
	• SP-01 • SP-02 • SP-03

Innerhalb dieser Kriterien wird Bezug auf die Festlegungen aus OIS genommen. Durch den Ausschluss eines großen Teils von OIS ist somit ein Einsatz von Kriterien aus dem Bereich SP nicht ohne weiteres möglich.

2.2.3 Personal (HR)

Das Personal (HR) verfolgt die Zielsetzung des "Sicherstellen(s), dass Mitarbeiter ihre Aufgaben verstehen, sich ihrer Verantwortung in Bezug auf Informationssicherheit bewusst sind und die Assets der Organisation bei Änderung der Aufgaben oder Beendigung geschützt werden" [12, S. 16].

Aufgrund des gewählten Szenarios, welches eine reine technische Vorkehrung zur Prävention von Angriffen behandelt, sind auch diese Kriterien nicht für eine Anwendung geeignet:

Tabelle 3 Anwendbare Kriterien aus dem Bereich HR

Anwendbare Kriterien	Nicht-anwendbare Kriterien	
	• HR-01 • HR-02 • HR-03	• HR-04 • HR-05 • HR-06

2.2.4 Asset Management (AM)

Die Zielsetzung des Asset Management (AM) verfolgt das "Identifizieren der organisationseigenen Assets gewährleisten und ein angemessenes Schutzniveau über deren gesamten Lebenszyklus sicherstellen" [12, S. 16].

Als Asset werden in diesem Kontext Objekte bezeichnet, die "während der Erstellung, Verarbeitung, Speicherung, Übermittlung, Löschung oder Zerstörung von Informationen benötigten Objekte im Verantwortungsbereich des Cloud-Anbieters, z.B. Firewalls, Loadbalancer, Webserver, Anwendungsserver und Datenbankserver." [12, S. 50] Hierbei kann nochmals in Hardware- und Software-Objekte unterteilt werden. Hardware-Objekte sind demnach physische und virtuelle Ressourcen, wie zum Beispiel Server, und Software-Objekte beschreiben Hypervisor, Container und Datenbanken [12, S. 50]. Da es sich bei dem gewählten Szenario um einen Angriff handelt, der speziell die Software-Objekte versiert, sind somit Kriterien aus dem Bereich AM eine gute Möglichkeit zur Bewertung der vorhandenen Konzepte.

Tabelle 4 Anwendbare Kriterien aus dem Bereich AM

Anwendbare Kriterien	Nicht-anwendbare Kriterien
• AM-01 • AM-02 • AM-06	• AM-03 • AM-04 • AM-05

2.2.5 Physische Sicherheit (PS)

Kernthema des Bereichs Nummer 5 Physische Sicherheit (PS) ist das "Verhindern von unberechtigtem physischen Zutritt und Schutz vor Diebstahl, Schaden, Verlust und Ausfall des Betriebs" [12, S. 16].

Aufgrund der Carve-Out Methode, durch die der Anbieter der Infrastruktur aus der Bewertung durch den kondensierten Katalog herausfällt, sind die Kriterien aus diesem Bereich nicht relevant. Sie behandeln die Sicherheitsvorkehrungen innerhalb des Gebäudes des Rechenzentrums und sind somit nicht Teil der Mechanismen für das SaaS-Produkt.

Tabelle 5 Anwendbare Kriterien aus dem Bereich PS

Anwendbare Kriterien	Nicht-anwendbare Kriterien	
	• PS-01 • PS-02 • PS-03 • PS-04	• PS-05 • PS-06 • PS-07

2.2.6 Regelbetrieb (OPS)

Innerhalb des Regelbetriebs (OPS) steht das "Sicherstellen eines ordnungsgemäßen Regelbetriebs einschließlich angemessener Maßnahmen für Planung und Überwachung der Kapazität, Schutz vor Schadprogrammen, Protokollierung und Überwachung von Ereignissen sowie den Umgang mit Schwachstellen, Störungen und Fehlern" im Mittelpunkt der Kontrolle [12, S. 16].

In diesem Szenario lässt sich zwischen Zuständigkeiten des SaaS-Anbieters und des Infrastruktur-Anbieters eine klare Linie ziehen. Aspekte wie die Sicherung und Wiederherstellung von Daten und die Härtung der verwendeten Komponenten, sind eindeutig im Zuständigkeitsbereich des Infrastruktur-Anbieters. Jedoch besteht der Regelbetrieb auch aus Themen wie der Protokollierung und Überwachung von Schwachstellen und die erforderliche Prüfung. Somit kann eine Teilmenge der Regularien für die Bewertung verwendet werden.

Dies ermöglicht eine Verwendung eines Großteils der Analysekriterien:

Tabelle 6 Anwendbare Kriterien aus dem Bereich OPS

Anwendbare Kriterien		Nicht-anwendbare Kriterien	
• OPS-10 • OPS-11 • OPS-12 • OPS-13 • OPS-14 • OPS-15 • OPS-16	• OPS-17 • OPS-18 • OPS-19 • OPS-20 • OPS-21 • OPS-22	• OPS-01 • OPS-02 • OPS-03 • OPS-04 • OPS-05 • OPS-06	• OPS-07 • OPS-08 • OPS-09 • OPS-23 • OPS-24

2.2.7 Identitäts- und Berechtigungsmanagement (IDM)

Mit Hilfe von Identitäts- und Berechtigungsmanagement (IDM) wird das "Absichern der Autorisierung und Authentifizierung von Benutzern des Cloud-Anbieters (in der Regel privilegierte Benutzer) zur Verhinderung von unberechtigten Zugriffen" gewährleistet [12, S. 16].

Dies spielt als Basis für die Handhabung von unautorisierten Zugriffen auf die Funktionalitäten des restlichen Systems eine bedeutende Rolle bei der Bewertung. Falls durch gezieltes Ausschalten

das Rechtevergabesystem nicht mehr funktionieren sollte, könnte ein Angreifer das gesamte System außer Gefecht setzen. Aus diesem Grund sind alle Basiskriterien in die Grundlage für die spätere Evaluation miteingeflossen.

Tabelle 7 Anwendbare Kriterien aus dem Bereich IDM

Anwendbare Kriterien		Nicht-anwendbare Kriterien
• IDM-01 • IDM-02 • IDM-03 • IDM-04 • IDM-05	• IDM-06 • IDM-07 • IDM-08 • IDM-09	

2.2.8 Kryptographie und Schlüsselmanagement (CRY)

Durch Kryptographie und Schlüsselmanagement wird das "Sicherstellen eines angemessenen und wirksamen Gebrauchs von Kryptographie zum Schutz der Vertraulichkeit, Authentizität oder Integrität von Informationen" gewährleistet [12, S. 16].

Die Gewährleistung der sicheren Übertragung von Daten spielt in diesem Szenario jedoch keine tragende Rolle und somit sind diese Bewertungskriterien nicht für die weitere Verwendung geeignet.

Tabelle 8 Anwendbare Kriterien aus dem Bereich CRY

Anwendbare Kriterien	Nicht-anwendbare Kriterien	
	• CRY-01 • CRY-02 • CRY-03	• CRY-04

2.2.9 Kommunikationssicherheit (COS)

Der Bereich der Kommunikationssicherheit (COS) widmet sich dem "Sicherstellen des Schutzes von Informationen in Netzen und den entsprechenden informationsverarbeitenden Systemen" [12, S. 17].

Neben den reinen Sicherheitsvorkehrungen innerhalb der Netze des Cloud-Anbieters werden hier auch technische Schutzmaßnahmen beschrieben, die z.B. im Fall von COS-01 Vorkehrungen zum Schutz vor Distributed-Denial-of-Service Attacken beschreiben.

Tabelle 9 Anwendbare Kriterien aus dem Bereich COS

Anwendbare Kriterien	Nicht-anwendbare Kriterien	
• COS-01	• COS-02 • COS-03 • COS-04 • COS-05	• COS-06 • COS-07 • COS-08

2.2.10 Portabilität und Interoperabilität (PI)

Das "Ermöglichen der Eigenschaft, den Cloud-Dienst über andere Cloud-Dienste oder IT-Systemen der Cloud-Kunden ansprechen zu können, die gespeicherten Daten bei Beendigung des Auftragsverhältnisses zu beziehen und beim Cloud-Anbieter sicher zu löschen" [12, S. 17] wird im Rahmen der Portabilität und Interoperabilität betrachtet.

Zusammenfassend handelt es sich hierbei um eine reine Dokumentation der vorhandenen Schnittstellen und die sichere Bereitstellung und Löschung von Daten. Für den Anwendungsfall der DoS-Angriffe sind diese Bewertungskriterien nicht relevant.

Tabelle 10 Anwendbare Kriterien aus dem Bereich PI

Anwendbare Kriterien	Nicht-anwendbare Kriterien
	• PI-01 • PI-02 • PI-03

2.2.11 Beschaffung, Entwicklung und Änderung von Informationssystemen (DEV)

Die Zielsetzung im Bereich der Beschaffung, Entwicklung und Änderung von Informationssystemen (DEV) ist das "Sicherstellen der Informationssicherheit im Entwicklungszyklus von Systemkomponenten des Cloud-Dienstes" [12, S. 17].

Die in diesem Kontext beschriebenen Kriterien erlauben die Qualität der Entwicklung der einzelnen Dienste zu bewerten. Letztlich wird geprüft, ob die entwickelten Features sicher gegenüber variablen Eingaben sind. Hier spielen zum Beispiel Angriffe wie die SQL-Injection eine tragende Rolle. Werden diese bösartigen Code-Schnipsel nicht vom System erkannt, bzw. als Plain-Text behandelt, ist es dem Angreifer möglich Daten in der Datenbank zu manipulieren, auszulesen oder im extremsten Fall zu löschen.

Es werden jedoch keine Entwicklungsarbeiten an dem SaaS-Produkt ausgelagert. Aus diesem Grund entfällt das Basiskriterium DEV-02.

Tabelle 11 Anwendbare Kriterien aus dem Bereich DEV

Anwendbare Kriterien		Nicht-anwendbare Kriterien
• DEV-01 • DEV-03 • DEV-04 • DEV-05	• DEV-06 • DEV-07 • DEV-08 • DEV-09 • DEV-10	• DEV-02

2.2.12 Steuerung und Überwachung von Dienstleistern und Lieferanten (SSO)

Die Steuerung und Überwachung von Dienstleistern und Lieferanten (SSO) konzentriert sich auf das "Sicherstellen des Schutzes von Informationen, auf die Dienstleister bzw. Lieferanten des Cloud-Anbieters (Subdienstleister) zugreifen können, sowie Überwachung der vereinbarten Leistungen und Sicherheitsanforderungen" [12, S. 17].

Die Überwachung und Bewertung von Dienstleistern, die zusätzliche Dienste zur Verfügung stellen und deren Überwachung, ist aufgrund des Carve-Out aller umliegenden Organisationen außer dem Anbieter der SaaS-Lösung ausgeschlossen. Diese Kriterien sind somit kein Bestandteil der Bewertung.

Tabelle 12 Anwendbare Kriterien aus dem Bereich SSO

Anwendbare Kriterien	Nicht-anwendbare Kriterien	
	• SSO-01 • SSO-02 • SSO-03	• SSO-04 • SSO-05

2.2.13 Umgang mit Sicherheitsvorfällen (SIM)

Mit dem "Gewährleisten eines konsistenten und umfassenden Vorgehens zur Erfassung, Bewertung, Kommunikation und Behandlung von Sicherheitsvorfällen" [12, S. 17] befasst sich der Bereich mit der Kennung „Umgang mit Sicherheitsvorfällen“.

Für die Implementierung einer allgemeinen Richtlinie für den Umgang mit Gefahren und Sicherheitsvorfällen bezüglich des Cloudproduktes sind diese Kriterien für eine Verwendung bei der Analyse geeignet. Sie ermöglichen Prozesse zu implementieren, die aus vergangenen Vorfällen neue Richtlinien für den Schutz des Produktes evaluieren.

Tabelle 13 Anwendbare Kriterien aus dem Bereich SIM

Anwendbare Kriterien	Nicht-anwendbare Kriterien
• SIM-01 • SIM-02 • SIM-03 • SIM-04 • SIM-05	

2.2.14 Kontinuität des Geschäftsbetriebs und Notfallmanagement (BCM)

Als Resultat der Domäne Kontinuität des Geschäftsbetriebs und Notfallmanagement steht das "Planen, Implementieren, Aufrechterhalten und Testen von Verfahren und Maßnahmen zur Kontinuität des Geschäftsbetriebs und für das Notfallmanagement" [12, S. 17].

Dies beschreibt somit im Groben die Handhabung von Sicherheitsvorfällen innerhalb des Rechenzentrums, die die Verfügbarkeit der Dienste beeinträchtigen könnten. Die Prävention dieser Gefahren liegt im Zuständigkeitsbereich des Infrastrukturanbieters.

Tabelle 14 Anwendbare Kriterien aus dem Bereich BCM

Anwendbare Kriterien	Nicht-anwendbare Kriterien
	• BCM-01 • BCM-02 • BCM-03 • BCM-04

2.2.15 Compliance (COM)

Innerhalb der Compliance (COM) steht das "Vermeiden von Verstößen gegen gesetzliche, regulatorische, selbstauferlegte oder vertragliche Anforderungen zur Informationssicherheit und [dem] Überprüfen der Einhaltung" [12, S. 17] im Mittelpunkt.

Dies betrifft größtenteils die Compliance bzw. die Einhaltung von Richtlinien im Informationsmanagementsystem (ISMS) aus dem Bereich OIS. Aus diesem Grund ist auch dieser Bereich nicht weiter für die nachfolgende Bewertung von Bedeutung.

Tabelle 15 Anwendbare Kriterien aus dem Bereich COM

Anwendbare Kriterien	Nicht-anwendbare Kriterien
	• COM-01 • COM-02 • COM-03 • COM-04

2.2.16 Umgang mit Ermittlungsanfragen staatlicher Stellen (INQ)

Das "Gewährleisten eines angemessenen Umgangs mit Ermittlungsanfragen staatlicher Stellen hinsichtlich juristischer Überprüfung, Information der Cloud-Kunden und Begrenzung des Zugriffs auf oder der Offenlegung von Daten" [12, S. 17] ist die Zielsetzung des Zuständigkeitsbereichs von Umgang mit Ermittlungsanfragen staatlicher Stellen.

Rechtliche Fragestellungen wie den Zugriff von staatlichen Stellen auf Daten von Cloud-Kunden werden in diesem Szenario nicht betrachtet. Aufgrund der Beheimatung der Dienste und deren Daten im Land des Kunden, sind Zugriffe von regierungsnahen Institutionen wie z.B. in Amerika in Folge des CLOUD Acts nicht in gleichem Maße zutreffend.

Tabelle 16 Anwendbare Kriterien aus dem Bereich INQ

Anwendbare Kriterien	Nicht-anwendbare Kriterien
	• INQ-01 • INQ-02 • INQ-03 • INQ-04

2.2.17 Produktsicherheit (PSS)

"Bereitstellen aktueller Informationen zur sicheren Konfiguration und über bekannte Schwachstellen des Cloud-Dienstes für Cloud-Kunden, geeigneter Mechanismen zur Fehlerbehandlung und Protokollierung sowie zur Authentisierung und Autorisierung von Benutzern der Cloud-Kunden" [12, S. 17] wird durch den letzten Bereich, die Produktsicherheit (PSS), garantiert.

Im Rahmen des letzten Bereiches, der Produktsicherheit, werden nochmals wichtige Kriterien für die Absicherung von Cloud-Diensten zusammengefasst, die über das ausgewählte Szenario hinaus Möglichkeiten für den sicheren Zugang der Cloud-Kunden sorgen.

Tabelle 17 Anwendbare Kriterien aus dem Bereich PSS

Anwendbare Kriterien		Nicht-anwendbare Kriterien
• PSS-01 • PSS-02 • PSS-03 • PSS-04 • PSS-05	• PSS-06 • PSS-07 • PSS-08 • PSS-09	• PSS-10 • PSS-11 • PSS-12

2.3 Kriterienkatalog für die STP Cloud

Auf Basis der vorangegangenen Untersuchung des Cloud Computing Compliance Criteria Catalogue – C5:2020 auf Verwendbarkeit der einzelnen Basiskriterien für die Beurteilung der Sicherheitsmechanismen im Falle des in Kapitel 2.1 ausgewählten Szenarios, konnten 51 Basiskriterien aus den 17 unterschiedlichen Bereichen isoliert werden.

Die nachfolgende Tabelle dient zur Beurteilung und zeigt den Grad der Erfüllung der einzelnen Kriterien nach Einschätzung bzw. der Analyse im Rahmen dieser Thesis. Die Ergebnisse der Analyse wurden durch Recherche von internen Dokumenten, Befragung von qualifiziertem Personal, hier der „Lenkungskreis Sicherheit", der im Rahmen der Entwicklung des Cloud-Produktes dessen Sicherheit überwacht und eigenen Untersuchungen der Anwendung, gewonnen.

Tabelle 18 Bewertungskriterien und Erfüllungsgrad für die Bewertung der STP Cloud

Beurteilungskatalog für die Erfüllung der ausgewählten Basiskriterien für ein vordefiniertes Szenario			
Szenario	Denial-of-Service Attacke		
Anzahl an selektierten Basiskriterien	50		
	Erfüllung durch die SaaS-Lösung und deren Anbieter		
Ref.	Erfüllt	Teilweise erfüllt	Nicht erfüllt
OIS-05	✗	✓	✗
AM-01	✗	✓	✗
AM-02	✗	✓	✗
AM-06	✗	✗	✓
OPS-10	✗	✓	✗
OPS-11	✗	✓	✗
OPS-12	✓	✗	✗
OPS-13	✗	✗	✓
OPS-14	✓	✗	✗
OPS-15	✓	✗	✗
OPS-16	✓	✗	✗
OPS-17	✓	✗	✗
OPS-18	✗	✗	✓
OPS-19	✗	✗	✓
OPS-20	✗	✗	✓
OPS-21	✓	✗	✗
OPS-22	✗	✗	✓
IDM-01	✗	✓	✗
IDM-02	✓	✗	✗
IDM-03	✗	✗	✓

IDM-04	✗	✓	✗
IDM-05	✗	✗	✓
IDM-06	✗	✓	✗
IDM-07	✗	✗	✓
IDM-08	✗	✓	✗
IDM-09	✓	✗	✗
COS-01	✗	✗	✓
DEV-01	✓	✗	✗
DEV-03	✗	✗	✓
DEV-04	✗	✗	✓
DEV-05	✗	✗	✓
DEV-06	✗	✓	✗
DEV-07	✓	✗	✗
DEV-08	✓	✗	✗
DEV-09	✓	✗	✗
DEV-10	✓	✗	✗
SIM-01	✗	✓	✗
SIM-02	✓	✗	✗
SIM-03	✗	✗	✓
SIM-04	✓	✗	✗
SIM-05	✗	✗	✓
PSS-01	✗	✗	✓
PSS-02	✗	✓	✗
PSS-03	✗	✗	✓
PSS-04	✗	✓	✗
PSS-05	✓	✗	✗
PSS-06	✓	✗	✗
PSS-07	✓	✗	✗
PSS-08	✓	✗	✗
PSS-09	✗	✓	✗
Σ	19	14	17

Die genauen Definitionen der Basiskriterien und die Begründung für die Erfüllung der Inhalte liegen in einem umfassenden Bewertungsdokument im Anhang der Thesis bei. Des Weiteren werden die Eigenschaften aufgezählt, die von der Anwendung und dem Anbieter nicht programmatisch umgesetzt oder in Form von dokumentierten Prozessen oder Richtlinien dem Kunden für den Umgang mit der SaaS-Lösung zur Verfügung gestellt werden.

Im nachfolgenden Schaubild wird zur Illustration die prozentuale Verteilung des Ergebnisses nochmals mit Hilfe eines Diagramms grafisch dargestellt.

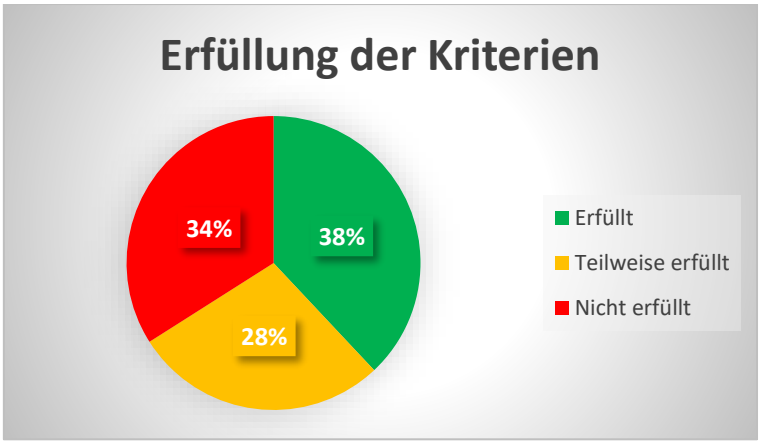

Abbildung 5 Grafische Darstellung mit prozentualer Verteilung des Erfüllungsgrades (Quelle: Eigene Abbildung)

2.4 Maßnahmen zur Optimierung des Ergebnisses

Basierend auf der Bewertung durch die zusammengestellten Basiskriterien aus dem Kapitel 2.3 sind Lücken bzw. unvollständige Konzepte im Rahmen der C5-Kriterien für das Betreiben der Cloud-Anwendung aufgetreten. Diese Möglichkeiten zur Optimierung des Erfüllungsgrades des Kriterienkatalogs werden mit Hilfe der nachfolgend formulierten Maßnahmen adressiert.

Die Ergänzungen zur Vervollständigung der teilweise bis nicht erfüllten Kriterien orientieren sich hierbei eng an den Vorgaben aus dem C5 und stellen somit ein Minimum (nach Ansicht des BSI) an vorhandenen Sicherheitskonzepten und Prozessen innerhalb des bewerteten Unternehmens bzw. der Anwendung dar. Hierbei liegt die Auslegung der einzelnen Maßnahmen und die Beispiele für eine Einsatzmöglichkeit bestimmter Werkzeuge auf den Annahmen und Empfehlungen des Autors und erhebt somit keinen Anspruch auf Vollständigkeit. Simultan gilt dies für den Umfang der vorangegangenen Analyse. Zur besseren Einordnung der einzelnen Maßnahmen in die jeweilig vordefinierten Bereiche, werden diese nach adressierter Thematik geordnet im Anschluss aufgeführt.

2.4.1 Organisation der Informationssicherheit (OIS)

Die aus dem Bereich OIS stammenden Kriterien (siehe OIS-05) empfehlen eine Etablierung eines (direkten) Kontaktes zu Organisationen, die sich mit der Problematik von Sicherheit für Anwendungen in der Cloud beschäftigen. Über diesen Kontakt können zukünftig aktuelle Informationen bezüglich Schwachstellen und Gefährdungen hinsichtlich der verwendeten Produkte und Technologien schneller ausgetauscht und somit in den aktuellen Entwicklungsprozess der Anwendung frühzeitig miteinfließen.

2.4.2 Asset Management (AM)

Im Rahmen der Bewertung hinsichtlich des Asset Management sind Dokumentationen der virtuellen Objekte wie z.B. der vorhandenen Microservices, aus denen die Anwendung besteht, zu führen. Darunter zählt auch eine Risikoeinschätzung dieser Güter. Solch eine Einschätzung ist nach Vorgaben des BSI auf der Brisanz der Daten, die von den Diensten in ihrer Funktion im System bearbeitet, gespeichert oder transferiert werden (siehe AM-01, AM-06) durchzuführen. Auf Basis der Risikoeinschätzung sind eine Klassifizierung und Kennzeichnung dieser Service-Objekte möglich, die nach dem Schutzbedarf der verarbeiteten Informationen die Objekte in unterschiedliche vorabdefinierte Schutzstufen einteilen. Durch Definition der Schutzstufen und Schutzbedarfe sind anschließend auch Themen wie die sichere Konfiguration der Anwendungen und den Umgang mit Anforderungen an Software- und Image-Versionen (hinsichtlich der Verwendung von Docker-Containern), sowie deren Aktualisierungen möglich (siehe AM-02).

2.4.3 Regelbetrieb (OPS)

Im Zusammenhang mit der Bewertung der OPS-Kriterien sind die nach dem BSI definierten Maßnahmen an einigen Stellen fraglich hinsichtlich ihres Einsatzes zur Steigerung der Sicherheit in der SaaS-Lösung. Darum werden Bemühungen im Hinblick auf die Vorgaben für das Aktivieren, Stoppen oder Pausieren der Protokollierung und dem Bereitstellen von Metadaten an den Cloud-Kunden nicht als sinnvoll erachtet. Im Kontext der Bewertung dieses Produktes wäre ein Stoppen der Aufzeichnungen der Aktivitäten in der Anwendung kontraproduktiv. Dies würde sich wiederum negativ in Bezug auf das Erkennen von Anomalien in den Anfragen und Protokolldaten auswirken und es würde eine Art „Black-Box“ entstehen. Aus diesem Grund werden beide Optimierungsmöglichkeiten nicht weiter von Seiten dieser Arbeit als mögliche Vorschläge betrachtet (siehe OPS-10, OPS-11).

Hingegen sind Konzepte wie ein automatisches Meldesystem von ungewöhnlichen Ereignissen bzw. Anomalien und die Weiterleitung von identifizierten Ereignissen an zuständiges Personal (siehe OPS-13) nicht im Rahmen der Anwendung vorhanden. Diesbezüglich sind weitere Maßnahmen wie Endpunkte (API's) für forensische Analysen von protokollierten Ereignissen (siehe OPS-15) Empfehlungen, die umgesetzt werden können.

Des Weiteren sind Dokumentationen für Prozesse und Anweisungen zur regelmäßigen (monatlichen) Identifikation von Schwachstellen, deren Beurteilung und Priorisierung für den späteren Entwicklungsprozess ein praktischer Leitfaden für die Entwickler darstellen, eine durchaus praktikable Grundlage für den weiteren Entwicklungsprozess. Mithilfe von Tools wie zum Beispiel dem OWASP ZAP Proxy [13] sind automatische Analysen von Web-Applikationen,

wie dieser SaaS-Lösung, auf bekannte Schwachstellen von Seiten der Web-Frameworks möglich. ZAP lässt sich entweder als eigenständige Anwendung in Form eines Docker-Containers in einer bestehenden Anwendung integrieren oder stellt ergänzend unterschiedliche APIs für die Verwendung in verschiedenen Programmiersprachen, u.a. C#, zur Verfügung (siehe OPS-18, OPS-22).

Ergänzend zu den automatisierten Analysen sind Penetrationstests von qualifiziertem internem oder externem Personal möglich, deren Ergebnisse und die daraus folgenden Nachbesserungen in definierten Zeiträumen durchzuführen sind (siehe OPS-19). Durch Etablieren eines Prozesses zur regelmäßigen Messung, Analyse und Bewertung der Verfahren zum Umgang mit Schwachstellen und Störungen ermöglicht dies in Kombination mit den automatischen Analysen einen Verfahrenskatalog, der die Aktualität der Anwendung hinsichtlich ihrer Sicherheitskonzepte kontinuierlich verbessert (siehe OPS-20).

2.4.4 Identitäts- und Berechtigungsmanagement (IDM)

Im Zuge des Rollen- und Rechtekonzeptes, sowohl Richtlinien zur Verwaltung von Zugangs- und Zugriffsberechtigungen für interne und externe Mitarbeiter des Cloud-Anbieters als auch Systemkomponenten (Services) im Autorisierungsprozess, sind basierend auf dessen Geschäfts- und Sicherheitsanforderungen Dokumentationen zur Vergabe von Benutzernamen, Zugangs- und Zugriffsberechtigungen und Funktionstrennungen empfehlenswert. Eine Komplettierung der bereits bestehenden Konzepte im weiteren Verlauf des Entwicklungsprozesses stellt für die Entwickler eine zusätzliche Möglichkeit zur Kontrolle der Rechtevergabe an maschinelle Clients dar (siehe IDM-01). Diese können auch Funktionen beschreiben, die zur Sperrung eines Nutzers bei Inaktivität oder mehrfach fehlgeschlagenen Anmeldeversuchen führen (siehe IDM-03). Zusätzlich wären Mechanismen denkbar, die Geo-Blocking ermöglichen. Somit besteht die Möglichkeit im Voraus Zugriffe aus bestimmten Ländern und Regionen, die für die Verwendung des Cloud-Diensts nicht gedacht sind, verweigert werden. Ein Angreifer müsste in diesem Fall sich zuerst über ein VPN oder Ähnliches Zugang in ein ausländisches Netz für einen gezielten Angriff auf dieses Produkt verschaffen.

Bezüglich der Prozesse zur Überwachung der Nutzerrechte nach Veränderung des Aufgabengebietes oder im Allgemeinen der regelmäßigen Überprüfung der zugeordneten Rechte, sind Konzepte zur Einführung dieser Prozesse empfohlen. Hiermit soll durch qualifiziertes Personal eine Rechteprüfung in regelmäßigen Abständen stattfinden. Bei Änderung von Aufgabengebieten z.B. von privilegierten Nutzern (z.B. spätestens 48 Stunden) oder allen anderen Änderungen bzw. Abweichungen (z.B. spätestens 7-14 Tage) soll nach Inkrafttreten eine

Anpassung im System durch das qualifizierte Personal vorgenommen werden (siehe IDM-04, IDM-05).

Eine automatische Überwachung für verdächtige Ereignisse, die durch Zugriff von privilegierten Nutzern entstehen könnte und deren Meldung, sind denkbare Abläufe, die innerhalb des Systems für die Sicherheit vor unbefugten Zugriffen von innen heraus sorgen (siehe IDM-06). Unter anderem die Alarmierung von Cloud-Kunden, auf deren Daten von Mitarbeitern des Cloud-Dienstes zugegriffen wurde, spielt in diesem Rahmen eine Rolle (siehe IDM-07).

In Anlehnung an die Dokumentationen bezüglich der Zugangs- und Zugriffsberechtigungen sind Aufzeichnungen hinsichtlich der Richtlinien zur Passwort-Vergabe und den getroffenen Konventionen, für das Erstellen eines Passworts und dessen sichere serverseitige Speicherung, sicherheitstechnisch relevant (siehe IDM-09). Eine Umsetzung von Seiten des Identity Providers durch Funktionen wie zum Beispiel die Validierung der Gültigkeit von 14 Tagen für ein initial vergebenes Passwort ist hier denkbar (siehe IDM-08). Auch Abweichungen von diesem Vorgehen sind für die spätere Entwicklung in Form einer Dokumentation relevant.

2.4.5 Kommunikationssicherheit (COS)

Eine Implementierung von technischen Schutzmaßnahmen zur Identifikation von anomalen Eingangs- und Ausgangs-Traffic-Mustern oder DDoS-Attacken sind innerhalb der Anwendung eine sicherheitskritische wie auch notwendige Funktion (siehe COS-01). Selbst bei Verwendung von Infrastrukturkomponenten wie zum Beispiel Firewalls oder Loadbalancer, ist eine hundertprozentige Erkennung von Angriffen nicht immer möglich. Jedoch können durch speziell konzipierte Dienste ein vom System selbst herbeigeführtes Schutzverhalten eingeleitet werden, das unabhängig vom Menschen und nicht erst durch Weiterleiten von Anomalien an übergeordnete Security Information- and Event Management-Systeme (SIEM) agiert.

2.4.6 Beschaffung, Entwicklung und Änderung von Informationssystemen (DEV)

Für die sichere Entwicklung des Cloud-Diensts sind Richtlinien und Anweisungen mit technischen und organisatorischen Maßnahmen mit den Schwerpunkten Sicherheit in der Software-Entwicklung (u.a. Anforderungen, Design, Implementierung, Tests und Überprüfungen), Sicherheit in der Softwarebereitstellung und die Sicherheit im Betrieb (Reaktion auf identifizierte Fehler und Schwachstellen) notwendig (siehe DEV-01). Mithilfe dieser Anforderungen können zusätzliche organisatorische Maßnahmen zur Verwaltung von Änderungen an den Diensten der SaaS-Lösung im Rahmen der Software-Bereitstellung beschrieben werden (siehe DEV-03). Ohne diese Regularien ist eine Risikobeurteilung von Änderungen auf Basis ihrer potenziellen

Auswirkungen auf die restlichen Dienste nicht durchführbar und eine entsprechende Kategorisierung ist nicht möglich (siehe DEV-05).

Nicht nur Dokumentationen bezüglich der oben genannten Regularien sind im Bereich DEV essenziell. Auch Programme zur Weiterbildung der internen Mitarbeiter bezüglich der Sicherheit in der Software-Entwicklung und Bereitstellung sind ein grundlegender Bestandteil für das Ausliefern sicherer Softwareprodukte (siehe DEV-04). Es wird hiermit eine regelmäßige und zielgruppenorientierte Sicherheitsausbildung und Sensibilisierung empfohlen.

2.4.7 Umgang mit Sicherheitsvorfällen (SIM)

Die Dokumentation, Kommunikation und Bereitstellung von Richtlinien und Anweisungen mit technischen und organisatorischen Maßnahmen sind im Umgang mit auftretenden Sicherheitsvorfällen für das betroffene Personal unerlässlich. Hier werden Vorgaben definiert, die die Klassifizierung, Priorisierung und Eskalation von Sicherheitsvorfällen beschreiben. Diese Prozesse reichen bis hin zur Verarbeitung von Vorfällen mit anschließender Information des Kunden über einen ihn betreffenden Zwischenfall (siehe SIM-01, SIM-03).

Ein Mechanismus zur Messung und Überwachung von Art und Umfang der Sicherheitsvorfälle und deren Meldung an die notwendigen Stellen wird ebenfalls nahegelegt (siehe SIM-05).

2.4.8 Produktsicherheit (PSS)

Unterstützend werden im Rahmen der Produktsicherheit Leitlinien bzw. Anleitungen definiert, die dem Kunden bei der sicheren Konfiguration des Cloud-Produktes helfen. Diese umfassen neben der sicheren Einrichtung, Informationsquellen zu bekannten Schwachstellen, Fehlerbehandlungs- und Protokollierungsmechanismen, Authentisierungsmechanismen, Rollen- und Rechtekonzepte (inkl. riskanter Rechte-Kombinationen) und Dienste und Funktionen zur Administration (siehe PSS-01).

Zur Absicherung der Qualität bezüglich der Sicherheit des Programmcodes werden dynamische u.a. auch statische Code-Analysen für den Einsatz und zur Überprüfung vor dem Ausrollen auf die Produktivumgebungen empfohlen. Deren Ergebnisse sollten anschließend nach Schweregrad der identifizierten Schwachstelle nach vordefinierten Kriterien beurteilt werden (siehe PSS-02). Hierzu verwendete Tools aus dem .NET Core Umfeld sind zum Beispiel „Security Code Scan - static code analyzer for .NET“, der auf Basis der OWASP Top Ten der meistausgenutzten Sicherheitslücken die Code Fragmente eines .NET-Projektes scannt [14]. Dies kann auch automatisch im Rahmen einer Continuous Integration und Continuous Delivery Pipeline geschehen. Beispielhaft für den Einsatz dieses Tools für Static Application Security Testing (SAST)

ist die Verwendung bei GitLab, dem direkten Konkurrenten von GitHub, in seiner eigenen CI/CD Toolbar [15].

Die somit identifizierten Schwachstellen, die nicht sofort behoben werden können, sind gemäß den Richtlinien nach einer Kategorisierung, wie zum Beispiel des Common Vulnerability Scoring System (CVSS) eingestuft und dem Kunden über einen Verweis auf ein Online-Register oder in Form einer Dokumentation ausgehändigt werden (siehe PSS-03, PSS-09).

Bezüglich der Protokollierung von Informationen des Cloud-Diensts ist die Verwendung hinsichtlich der Schaffung besserer Sicherheitsstandards fraglich. Hier wird die Möglichkeit einer Einsicht des Kunden in die protokollierten Daten des Dienstes angestrebt. Mithilfe dieser Daten soll er anhand von Fehlerbehandlungsmechanismen auftretende Störungen selbst beheben können. Diese Funktion wird jedoch von Seiten eines vom Cloud-Anbieter betriebenen Support-Centers übernommen, das technische wie auch logische Problematiken entgegennimmt und als Teil des Service-Pakets für den Kunden bearbeitet. Aus diesem Grund wird diese Funktion nicht als Teil der Optimierung verstanden. Sie wird hier anderweitig abgedeckt.

2.5 Analyse der vorhandenen Sicherheitskonzepte

In den nachfolgenden Kapiteln wird der aktuelle Stand an Sicherheitskonzepten der Software-as-a-Service Lösung, auch „LEXolution.FLOW“ genannt, analysiert und einem möglichen Soll-Zustand gegenübergestellt. Dieser Soll-Zustand orientiert sich hierbei an den aus Kapitel 2.4 benannten Maßnahmen zur Optimierung der bestehenden Sicherheitskonzepte. Explizit wird hierbei auf die technischen Schutzmaßnahmen der Cloud-Lösung eingegangen. Diese werden als Basis für die spätere Implementierung eines Mechanismus zum Schutz der Anwendung gelegt.

2.5.1 Ist-Zustand

Die bewertete SaaS-Lösung der STP wird über einen Cloud Service Provider (nachfolgend auch CSP) zur Verfügung gestellt, bedeutet die notwendige Hardware wird nicht In-House betrieben, sondern extern in Bezug auf den Bedarf hinzugebucht. Hierbei handelt es sich um ein deutsches Rechenzentrum, welches seine Daten ausschließlich in Standorten innerhalb von Deutschland speichert. In Folge des Beschlusses der amerikanischen Regierung ist es mittels des CLOUD Acts (Clarifying Lawful Overseas Use of Data) regierungsnahen Institutionen, wie zum Beispiel der NSA, CIA oder FBI, möglich sich ohne Einwilligung oder vorheriges Informieren der Cloud Nutzer Zugang zu den gespeicherten Daten der Cloud Provider zu verschaffen [16]. So wäre ein Hosting bei Microsoft (Azure), Amazon Web Services (AWS) oder Google Cloud Platform (GCP) mit der deutschen Rechtslage für die Zielgruppe der STP im Rahmen des Berufsträgergeheimnis § 203 StGB nicht vereinbar [17].

Bei der Analyse des Ist-Zustandes steht jedoch nicht die Bewertung der Regularien zum Betreiben einer Cloud Lösung im bzw. die Sicherheitskriterien für das Rechenzentrum im Mittelpunkt, sondern die Resistenz vor gängigen Angriffen auf Cloud Softwarelösungen. Hiermit wird eine Grenze zwischen der Software und der Plattform beziehungsweise der darunterliegenden Infrastruktur gezogen (Carve-Out Methode).

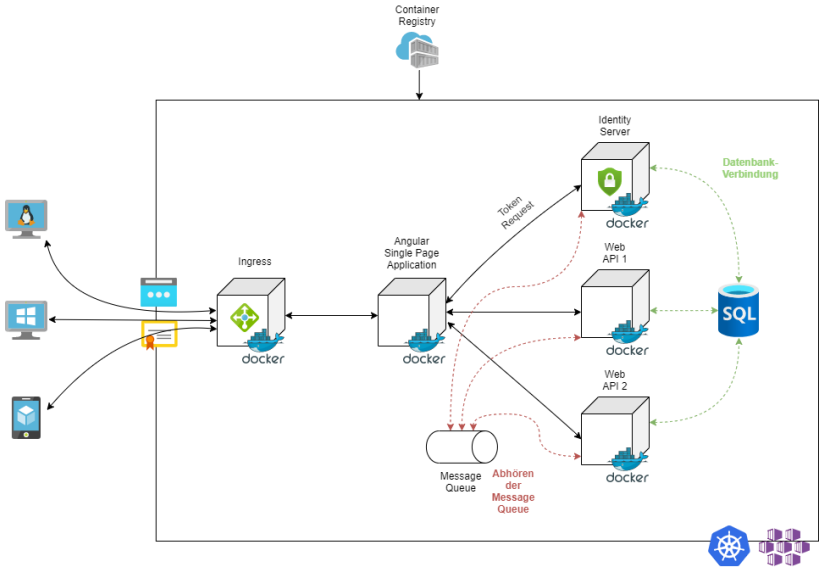

Abbildung 6 Vereinfachte Darstellung der STP Cloud SaaS-Lösung (Quelle: Eigene Abbildung)

Mithilfe der vorangegangenen vereinfachten Darstellung der STP Cloud Lösung soll nun die Bestandsaufnahme des Ist-Zustandes erfolgen. Der Anwendungsverbund besteht im übergeordneten Sinne aus unterschiedlichen Single Page Applikationen (nachfolgend auch SPA genannt), die die grafische Benutzeroberfläche der Anwendung repräsentieren. Hierbei wird je nach gewünschtem Menü über den vorgelagerten Ingress im Kubernetes Kluster die entsprechende SPA angesprochen. Somit verteilt sich die Last nicht nur auf eine einzelne Webanwendung. Über die Weboberfläche wird dem Kunden die Eingabe von für den Prozess relevanten Daten ermöglicht. Die Verarbeitung und Speicherung der Daten wird im Anschluss über Web Programmierschnittstellen (nachfolgend auch API genannt) gewährleistet. Die Implementierung der APIs ist mit dem .NET Framework und C# als Programmiersprache umgesetzt worden. Für die sichere Übertragung der Daten vom Frontend an das Backend wird mittels TLS 1.2 der ein- und ausgehende Datenstrom verschlüsselt. Dies verhindert im ersten Schritt das ungewünschte Mitschneiden und -lesen von Datenpaketen. Bezüglich der Verwaltung von Rechten wird im Backend ein Identity Provider (hier IdentityServer in der Version 4) bereitgestellt. Hierbei handelt es sich um einen Service, der das OpenID Connect und das OAuth2 Protokoll umsetzt. Die Nutzung dieses Diensts gewährleistet eine zentrale Nutzerverwaltung für

alle Tenants und die Vergabe von Rechten unterschiedlicher Granularität. Bevor der Nutzer Zugriff auf die SaaS-Lösung erhält, muss er sich initial beim Identity Provider authentifizieren. Stimmen Nutzername und Passwort mit den hinterlegten Anmeldedaten überein, erhält der Nutzer, die ihm zugewiesenen Rechte, er wird somit autorisiert. Im Hintergrund des Anmeldeprozesses findet der Authorization Code Flow with Proof-Key-of-Code-Exchange (nachfolgend auch Authorization Code Flow + PKCE genannt) statt. Hierbei handelt es sich um das aktuellste und sicherste im OAuth2 Protokoll spezifizierte Verfahren für den Austausch von Tokens zwischen Identity Providern und clientseitigen Webanwendungen, wie zum Beispiel SPAs.

Für die Bewertung des aktuellen Sicherheitszustandes zur Erkennung und Abwendung von DoS-Angriffen wurden mithilfe Oracle Virtualbox in der Version 6.1 [18], einer virtuellen Maschine und dem Betriebssystem Kali Linux in der Version 2021.1 [19], welches in der Industrie für Penetrationtesting von Business Anwendungen verwendet wird, im Rahmen dieser Thesis von Seiten des Autors gezielt Angriffe auf das System ausgeführt. Die Ausführung der Angriffe wurde in zwei Phasen durchgeführt. Die erste Angriffsphase erstreckte sich vom 01.06.2021 bis zum 04.06.2021, in der die Tools auf dem oben genannten Zielsystem installiert und prototypisch für kurze Zeit verwendet wurden. In der zweiten Phase vom 07.06.2021 bis zum 11.06.2021 wurden nochmals gezielte Services und Endpunkte angegriffen. Hierbei wurden die Open Source DoS- bzw. DDoS-Tools Low Orbit Ion Cannon (LOIC) [20], GoldenEye [21] und Slowloris [22] verwendet. Die Verwendung dieser Tools begründet sich auf dem weitverbreiteten Einsatz in der Community und der Referenz aus dem Paper „Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization“ [23], aus dem das verwendete Dataset für das maschinelle Lernmodell in Kapitel 2.6.3.3 stammt. Hiermit soll gewährleistet werden, dass der erzeugte Netzwerkverkehr, der in den späteren Versuchen im Rahmen des programmatischen Teils erzeugt wird, auch wieder durch das Modell erkannt werden und somit ein Vergleich gezogen werden kann, ob das maschinelle Lernmodell funktionsfähig ist. Die hieraus gewonnenen Erkenntnisse werden nun in den nachfolgenden Kapiteln genauer beleuchtet.

2.5.1.1 DDoS-Angriffsszenario mit LOIC

Die Testreihe wurde mit dem Tool LOIC mit dem Ziel auf den Endpunkt des Identity Providers eingeleitet. Die Low Orbit Ion Cannon ist ein in C# geschriebenes Desktop-Programm, das in zwei unterschiedlichen Modi zur Verfügung steht. Es funktioniert einerseits im sogenannten „Manual Mode“, in dem nur eine Maschine (auf der das Programm läuft) mithilfe von Thread-Pools Anfragen an einen bestimmten Endpunkt oder an unterschiedliche Endpunkte des Services schicken kann. Für einen fortgeschrittenen DDoS-Angriff verfügt es auch über den „HiveMind“. Der „HiveMind“ erlaubt die Verbindung mit einem fremden Internet Relay Chat-Server (IRC),

der remote von einem Administrator gesteuert wird und alle angeschlossenen LOIC als Bot-Netz operationalisiert. Hiermit hat der Endanwender, auf dessen System die LOIC keinerlei Kontrolle über dessen Verhalten mehr und kann Teil eines größer angelegten Angriffes werden. Innerhalb dieses Szenarios wurde jedoch nur der „Manual Mode“ verwendet, da kein eigener IRC-Server zur Verfügung stand und die Verbindung auf unbekannte Server als zu risikoreich eingestuft wurde. Nach dem Start der Anwendung begann diese den festgelegten Endpunkt mit einer Flut an Anfragen zu kontaktieren. Die Besonderheit dieser Anfragen ist, dass es sich hierbei um keine Keep-Alive Anfragen handelt, wie zum Beispiel beim Tool Slowloris, sondern hier wird versucht durch die immense Masse an Requests das System zum Überlasten zu bringen.

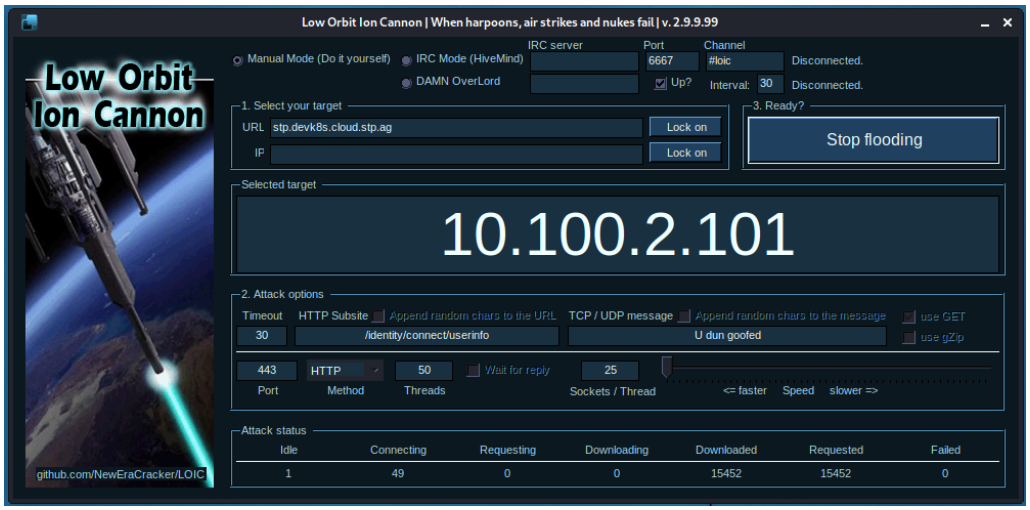

Abbildung 7 Angriff mit der LOIC auf den Login-Endpunkt des Identity Providers
(Quelle: Eigene Abbildung)

Das Ergebnis des ersten Testlaufs war, dass der NGINX Ingress, der als Reverse Proxy an der Schnittstelle zwischen Kluster und dem Internet steht, die Anfragen nicht an den gewünschten Service weiterleitete. Wie im nachfolgenden Ausschnitt aus der Log-Datei des NGINX kenntlich wurde, hat er die eingehenden Requests mit dem Status Code „**400 Bad Request**“ gekennzeichnet und nicht in das dahinterliegende Kluster geleitet.

```
10.100.110.51 - - [09/Jun/2021:15:54:06 +0200] "GET /identity/connect/userinfo HTTP/1.1" 400 280 "-" "Mozilla/5.0 (Windows NT 6.2; WOW64; r
v:37.0) Gecko/20100101 Firefox/37.0"
10.100.110.51 - - [09/Jun/2021:15:54:06 +0200] "GET /identity/connect/userinfo HTTP/1.1" 400 280 "-" "Mozilla/5.0 (Windows NT 6.0; WOW64; r
v:45.0) Gecko/20100101 Firefox/45.0"
10.100.110.51 - - [09/Jun/2021:15:54:06 +0200] "GET /identity/connect/userinfo HTTP/1.1" 400 280 "-" "Mozilla/5.0 (Windows NT 6.2; WOW64; r
v:38.0) Gecko/20100101 Firefox/38.0"
10.100.110.51 - - [09/Jun/2021:15:54:06 +0200] "GET /identity/connect/userinfo HTTP/1.1" 400 280 "-" "Mozilla/5.0 (Windows NT 6.0; rv:37.0)
 Gecko/20100101 Firefox/37.0"
10.100.110.51 - - [09/Jun/2021:15:54:06 +0200] "GET /identity/connect/userinfo HTTP/1.1" 400 280 "-" "Mozilla/5.0 (Windows NT 10.0; rv:37.0
) Gecko/20100101 Firefox/37.0"
10.100.110.51 - - [09/Jun/2021:15:54:07 +0200] "GET /identity/connect/userinfo HTTP/1.1" 400 280 "-" "Mozilla/5.0 (Windows NT 10.0; WOW64;
rv:38.0) Gecko/20100101 Firefox/38.0"
10.100.110.51 - - [09/Jun/2021:15:54:07 +0200] "GET /identity/connect/userinfo HTTP/1.1" 400 280 "-" "Mozilla/5.0 (Windows NT 6.2; WOW64; r
v:37.0) Gecko/20100101 Firefox/37.0"
10.100.110.51 - - [09/Jun/2021:15:54:15 +0200] "GET /identity/connect/userinfo HTTP/1.1" 400 280 "-" "Mozilla/5.0 (Windows NT 10.0; rv:39.0
) Gecko/20100101 Firefox/39.0"
```

Abbildung 8 Access Log-Datei des NGINX Reverse Proxy vor dem Dev-Kluster
(Quelle: Eigene Abbildung)

Dieses Verhalten wurde im späteren Verlauf der Testreihe an der selbstkonzipierten Testumgebung ebenfalls beobachtet. Somit wurde die dahinterliegende Anwendung nicht von dem Angriff getroffen und es wurde kein „Denial-of-Service“ herbeigeführt.

Die Analyse in der lokalen Testumgebung ergab folgendes Ergebnis. Mittels LOIC werden „Plain HTTP Requests“ verschickt. Durch einen Angriff auf Port 443, also HTTPS, des Reverse Proxy werden diese Anfragen von Seiten des Proxy somit stets geblockt, da es sich um keine auf TLS-Basis verschlüsselten Anfragen handelt. Diese Information konnte durch das Erhöhen des Logging-Levels von „Info“ auf „Debug“ innerhalb der „nginx.conf“-Datei und eines erneuten Testlaufes gewonnen werden.

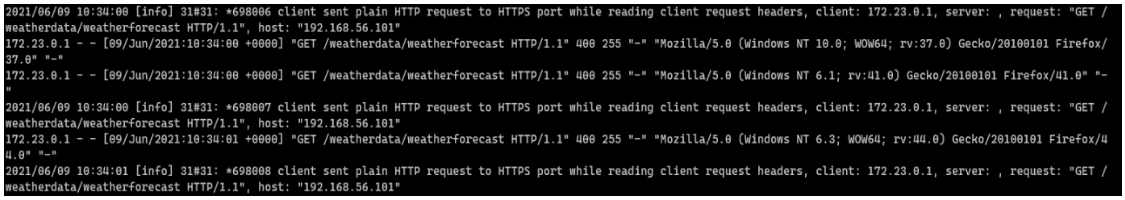

Abbildung 9 Ergebnis des erhöhten Logging-Levels in Kombination mit Angriff von „LOIC“ auf Port 443 (Quelle: Eigene Abbildung)

Nach der Anpassung des Ports von 443 auf den HTTP Port 80 hat der NGINX in der lokalen Testumgebung die Anfragen an das entsprechende Backend weitergeleitet. Dasselbe Verhalten konnte nicht innerhalb des Development Klusters der STP beobachtet werden.

Hier wurde der Endpunkt des Identity Providers `/identity/connect/userinfo` angesprochen. Dieser stellt der Anwendung Informationen über den angemeldeten Nutzer gegen den Austausch eines Access Tokens zur Verfügung. Die Wahl dieses Endpunktes basierte auf der Abfrage des übergeordneten Endpunktes `/identity/.well-known/openid-configuration`. Hierbei handelt es sich um den „Discovery Endpoint“ des Identity Providers. Dieser ist nach Außen ohne vorherige Authentifizierung zugänglich und ermöglicht dem Angreifer spezifische Endpunkte des Identity Providers zu identifizieren und mit DoS- oder DDoS-Angriffen zu überlasten.

Bezüglich der fehlgeschlagenen Versuche über Port 80 Anfragen an diesen speziellen Endpunkt zu schicken, ist anzunehmen, dass dieser Port für Anfragen in das Development Kluster deaktiviert wurde. Diese Tatsache konnte nach einem Blick in die Konfigurationsdatei im NGINX Reverse Proxy unter den `sites-enabled` bestätigt werden. Der Angriff von LOIC gegen die SaaS-Lösung blieb somit ohne Erfolg.

2.5.1.2 DoS-Angriffsszenario mit Slowloris

Im Anschluss an die Testreihe aus Kapitel 2.5.1.1 kam das Tool Slowloris zum Einsatz. Bei Slowloris handelt es sich um ein in Python geschriebenes Konsolen-Programm, welches 150

Sockets auf dem Client öffnet und mit Keep-Alive Anfragen die Verbindung zum Ziel aufrechterhält. Dies funktioniert über das wiederholte Senden von kleinen Paketen an den entsprechenden Endpunkt. Somit wird die Verbindung vom Ziel aus nicht abgebaut und der Ressourcen-Pool mit der Zeit erschöpft.

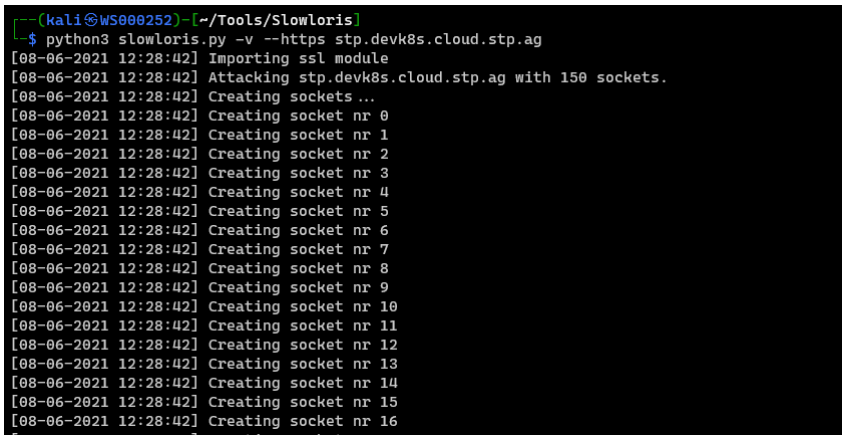

Abbildung 10 Angriff mittels „Slowloris" auf den NGINX Reverse Proxy
(Quelle: Eigene Abbildung)

Auch dieser Angriffsvektor verlief ohne Erfolg. Der Reverse Proxy hat die Anfragen allesamt mit dem HTTP Status Code „**408 Connection Timeou**t" gekennzeichnet. Dies deutet darauf hin, dass die eingebettete Zeit für das Aufrechterhalten von Verbindungen in der Standard-Konfiguration des Proxies deutlich kürzer ist. Somit müssten von Seiten des Angreifers Änderungen am Tool vorgenommen werden, um diesen Angriff erfolgreich durchzuführen. Da dieser aber keine Möglichkeit zur Einsicht in die Log-Dateien des Proxies besitzt, wird der Angriff von Seiten des Proxies bereits abgewehrt und der „Denial-of-Service" bleibt aus.

Abbildung 11 Log-Datei nach dem Angriff mit „Slowloris"
(Quelle: Eigene Abbildung)

2.5.1.3 DoS-Angriffsszenario mit GoldenEye

Die Testreihe wurde mit der Angriffs-Simulation des Tools GoldenEye abgeschlossen. Bei GoldenEye handelt es sich, wie bei Slowloris, um eine Konsolen-Applikation, die in Python geschrieben worden ist. Über vorgegebene Kommandozeilen-Argumente lassen sich verschiedene Parameter während der Ausführung des Programms anpassen. Von der Funktionsweise ähnelt es

der LOIC und kann unterschiedliche Arten von HTTP Anfragen an ein gewünschtes Ziel schicken.

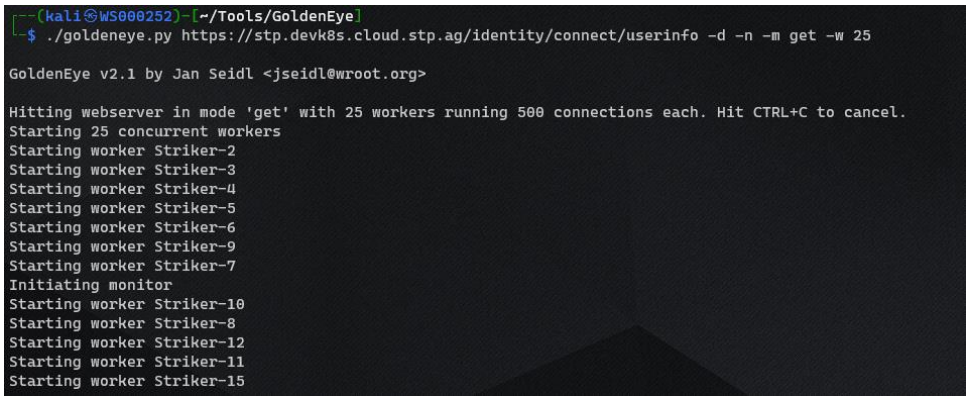

Abbildung 12 Angriff mit dem Tool „GoldenEye" auf die Dev-Cloud
(Quelle: Eigene Abbildung)

Mithilfe dieses Tools konnte der gewünschte Endpunkt durch den Proxy hinweg angesprochen werden. Die Log-Dateien aus dem NGINX und dem IndentityServer wiesen die Vielzahl an produzierten Anfragen auf. Hierbei wurden von Seiten des NGINX wie auch von Seiten der dahinterliegenden Anwendung keinerlei Maßnahmen ergriffen, um diesen kontrollierten Angriff in einer Art einzudämmen, um eine spätere Überlastung auszuschließen.

Abbildung 13 Log-Nachrichten während des Angriffs mit "GoldenEye" auf den IdentityServer
(Quelle: Eigene Abbildung)

2.5.2 Soll-Zustand

Resultierend aus der Analyse des Ist-Zustandes und der Bewertung mithilfe des Kriterienkatalogs und den daraus abgeleiteten Maßnahmen ist für den Soll-Zustand der SaaS-Lösung folgendes Konzept denkbar. Aufgrund des Fehlens eines internen Kontrollmechanismus für das Erkennen von netzbasierten Angriffen und Anomalien in Eingangs- und Ausgangs-Traffic-Mustern (siehe 2.4.5 Kommunikationssicherheit (COS)), wäre die Konzeption eines Dienstes mit solch einer Funktion als zusätzlicher Sicherheitsaspekt für den Anwendungsverbund geeignet.

Die Reaktion der Anwendung auf ein wiederholtes unautorisiertes Anfragen von öffentlich identifizierbaren Endpunkten ist wie zu erwarten ein Status-Code „**401 Unauthorized**", auf den keine weitere Maßnahmen in diesem Prozessmodell folgen.

Sollte hier ein Angreifer mittels eines Denial-of-Service oder des fortgeschritteneren Distributed-Denial-of-Service Angriffs versuchen das System über das Fluten von Endpunkten mit Anfragen zu überlasten, wird die Anwendung entweder auf Basis ihrer darunter liegenden Plattform, wie in diesem Fall Kubernetes, dementsprechend Skalieren oder unter der Last zusammenbrechen. Als Folge entstehen Kosten für den Anbieter und Ausfallzeiten für den Kunden.

Nachfolgend soll nun auf Basis der vorliegenden Architektur und evaluierten Prinzipien zum Schutz von Software-as-a-Service-Lösungen ein mögliches Modell vorgestellt werden, das durch seine Funktion eine Lösung für diese Problematik darstellen könnte.

Auf Basis der Konfigurationen, die innerhalb des NGINX möglich sind, um einen DoS- oder DDoS Angriff abzumildern, ist nun die Frage, ob es möglich ist ein selbstagierendes System mit den Funktionen eines NGINX Reverse Proxy zu konzipieren. Bei NGINX müssen zum Beispiel gezielte Endpunkte mit Maßnahmen für das Gewährleisten von einer bestimmten Anzahl an zulässigen Anfragen („Throttling“) pro Nutzer manuell und per Hand durch den Administrator in die einzelnen Server-Blöcke in der Konfigurationsdatei eingegeben werden. Auch das Blockieren und Erlauben von bestimmten IP-Adressbereichen („IP-Blocking“), die sich als mögliche Angreifer durch die manuelle Analyse der Log-Dateien ergeben haben, muss per Hand erfolgen. Es gibt auf Basis des proprietären Produktes NGINX Plus Möglichkeiten über das Buchen von bestimmten Modulen diese Funktionen dynamisch in das Produkt einzubinden, jedoch sind diese Erweiterungen nur in dieser Version des Reverse Proxy zugänglich. Die freierhältliche Open Source Variante, die auch hier zum Einsatz kommt, muss durch eigene Konfigurationen mittels der Programmiersprache Lua oder anderen frei erhältlichen Modulen je nach Nutzen angepasst werden.

Ziel der nachfolgenden Kapitel wird es sein, solch eine Anwendung zu konzipieren und im Anschluss umzusetzen. Hierbei wird der Schwerpunkt auf das Erkennen dieser speziellen Art von Angriffen und das Throttling der hiermit verbundenen Anfragen gelegt.

2.6 Erkennung und Prävention von Gefahren

Speziell für das Szenario des Denial-of-Service oder der weitaus fortgeschritteneren Variante des Distributed-Denial-of-Service Angriffs besteht das Ziel darin eine öffentlich erreichbare Webanwendung mittels einer immensen Masse an Anfragen so lange zu fluten, bis die Last an Anfragen das System zum Zusammenbruch bringt. Dies zu verhindern, wird im Rahmen des zweiten Teils der Thesis die Aufgabe eines eigenentwickelten Dienstes.

Das Konzept basiert darauf, dass der Dienst eine Art Reverse Proxy simuliert, der in seiner Funktionsweise gängigen Lösungen wie zum Beispiel dem NGINX Reverse Proxy [24] ähnelt. Jedoch wird dieser Service auf Basis eines .NET [25] implementierten Microservices aufbauen, der bei Erkennen von Anomalien (hier die DoS-Angriffe) die dahinterliegenden Services durch ein Throttling (Verlangsamen) der Anfragen vor einer Überlastung und somit eines Ausfalls schützt. Die zugrundeliegende Logik, ob es sich um solch eine Art Angriff handelt, wird mithilfe eines maschinellen Lernmodells erfolgen, das mit Anfragenmustern trainiert wurde, die diese Angriffe enthalten. Sollte es dem Modell möglich sein diese Anomalien zu erkennen, kann es mittels ML.NET [26], einer jungen Technologie aus dem Hause von Microsoft, in den Microservice integriert und dort das Throttling einleiten. Die Funktionsweise eines Proxies wird dem Service über das Open-Source Projekt YARP [27](„Yet Another Reverse Proxy“) verliehen, welches auch aus dem Hause Microsoft stammt. Die Kombination dieser beiden Technologien stellt in diesem Kapitel die Herausforderung bei der Programmierung dieses zusätzlichen Sicherheitskonzeptes dar.

2.6.1 Konzeption einer Testumgebung

Für die Evaluation und Umsetzung des ML.NET Proxy (nachfolgend auch ML.Proxy genannt) Dienstes wurde anhand der in Kapitel 2.5.1 beschriebenen Architektur der Anwendung eine beispielhafte Testumgebung erstellt. Diese besitzt die grundlegenden Eigenschaften des Originals und kann somit als Substitut für Testzwecke verwendet werden, ohne auf die eigentliche Anwendungslandschaft zugreifen zu müssen.

Auf Basis des Originals besteht dieser Anwendungsverbund aus einem NGINX Reverse Proxy in der Version 1.21.0, der den im Kubernetes Kluster befindlichen NGINX Ingress auf der lokalen Testumgebung simuliert. Er übernimmt die Weiterleitung der Anfragen an die dahinterliegenden Services und führt ein Load Balancing durch. Hinter dem Proxy ist eine Single Page Applikation als Frontend für die Testanwendung geschalten. Sie wurde mit dem Web-Framework Angular [28] von Google in der Version 8 implementiert und dient dem Authentifizieren und Autorisieren eines Nutzers und der damit verbundenen Datenabfrage an den übrigen drei Microservices. Für die Authentifizierung und Autorisierung kommt der IdentityServer [29] als OpenID Connect und OAuth2 Provider zum Einsatz. Hierbei handelt es sich um ein Framework, welches die genannten Standards umsetzt und als Identity Provider (IP) eingesetzt werden kann. Hiermit lassen sich gezielt einzelne Anwendungen absichern und der Zugriff von maschinellen Clients wie auch Usern der Anwendung durch Vergabe von eigendefinierten Rechten steuern. Hierzu wurde der

IdentityServer als NuGet-Package in eine .NET Core [25] Webanwendung eingebunden. Die restlichen beiden Microservices stellen Dummy-Services zur Abfrage von Standort- und Wetterdaten dar. Beide Services wurden mit dem IP verknüpft, sodass nur autorisierte Nutzer mit einem gültigen JWT die vorgehaltenen Daten abfragen können.

Neben einem lokalen Betreiben dieser Anwendung mittels Docker Compose [30]. Wie im nachfolgenden Schaubild exemplarisch dargestellt und für die spätere Simulation der Produktivbedingungen wird die Anwendung auf einem in Microsoft Azure Kubernetes Service (AKS) betriebenen Kubernetes [31] Kluster ausgerollt. Auf die Nutzung einer Message Queue und einer relationalen Datenbank für die Persistierung der Daten wurde verzichtet. Der Fokus wird ausschließlich auf die Manipulation bzw. das Throttling der Anfragen an das Backend gelegt.

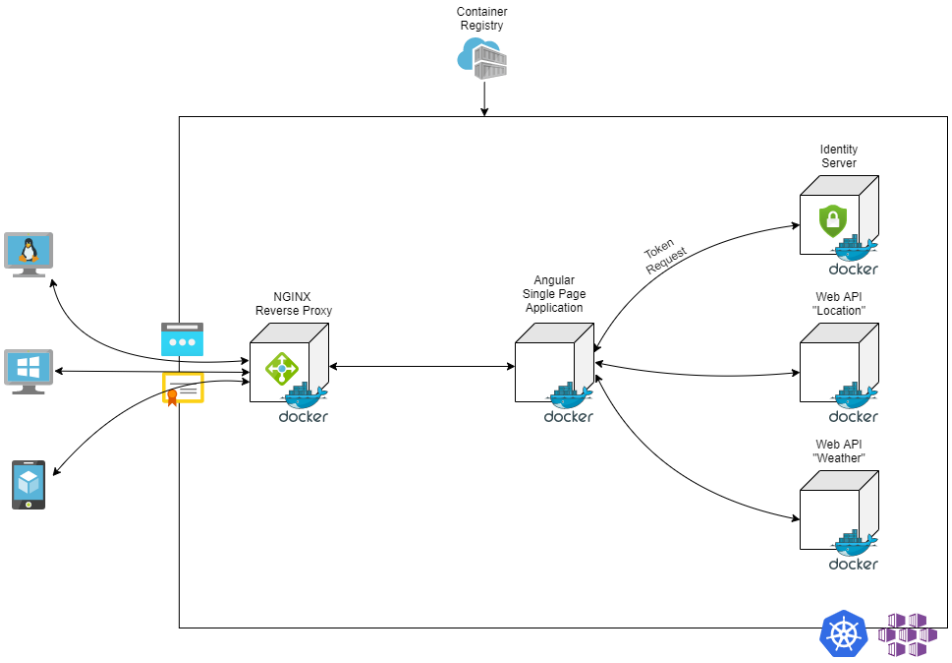

Abbildung 14 Testumgebung für die Simulation der realen Anwendungslandschaft (Quelle: Eigene Abbildung)

2.6.2 Mögliche Architektur mit ML.NET Proxy Service

Durch die Nutzung des ML.NET Proxy Service soll auf Auffälligkeiten in der Historie bzw. der aktuellen Anfragen von Außerhalb reagiert werden. Hierzu muss der Proxy jedoch zwischen Backend und dem Gateway des Klusters platziert werden. Eine mögliche Beispiel-Architektur ist im nachfolgenden Schaubild dargestellt.

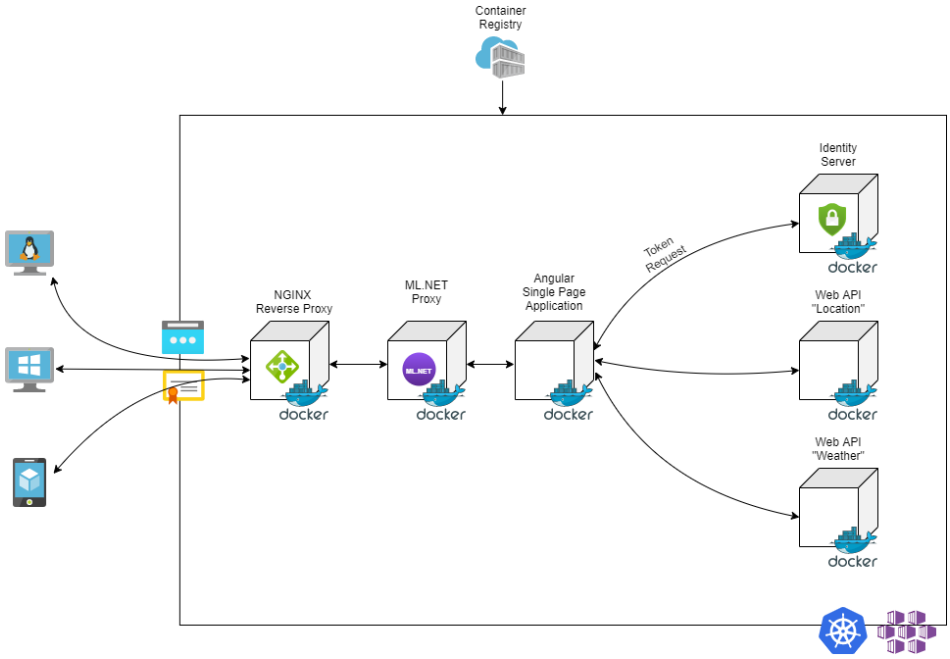

Abbildung 15 Beispiel-Architektur mit zusätzlichem Service für die Prüfung der eingehenden Requests mit Möglichkeit zum Throttling (Quelle: Eigene Abbildung)

2.6.3 Vorgehen bei der Implementierung

Die Entwicklung des kognitiven Dienstes zur Erkennung von DoS-Angriffen wird im Rahmen eines inkrementellen Prozesses durchgeführt. Arbeitspakete werden jeweils für zwei Wochen ausgerichtet, um ein frühzeitiges Erkennen von Problemen und Inkompatibilitäten zu identifizieren und die Entwicklung auf einen anderen Teilaspekt zur Absicherung der Anwendung zu fokussieren.

2.6.3.1 Auswahl des Datensatzes für das Training des ML-Modells

Innerhalb des ersten Arbeitspaketes im Zeitraum vom 14.06.2021 bis zum 25.06.2021 stand die Auswahl und Evaluierung eines Datensatzes für das Trainieren des Machine Learning (ML) Modells im Mittelpunkt. Hierbei wurden verschiedene Datasets („Datensätze“) hinsichtlich ihrer Aktualität und der Verwendbarkeit geprüft. Hierbei fiel die Auswahl auf das Dataset CSE-CIC-IDS2018, welches in Zusammenarbeit zwischen der Communication Security Establishment (CSE), der kanadischen nationalen Agentur für Kryptographie, und der University of New Brunswick entstand. Das CSE ist stellt hierbei eine regierungsnahe Institution dar, die die Versorgung der Regierung mit Informationen hinsichtlich Informationssicherheit und Aktionen ausländischer Geheimdienste im kanadischen Teil des Netzes [32] sicherstellen soll. Die Erstellung des oben genannten Datensatzes ging mit der gleichnamigen Arbeit „Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization“ [23] einher, die die Aktualität und Eignung der bisherigen öffentlichen Datensätze, wie zum Beispiel DARPA [33], KDD [34] und DEFCON [35] usw. prüfte und abschließend selbst ein maschinelles Lernmodell konzipierte,

um die verwendeten Angriffe innerhalb der Log-Nachrichten zu erkennen. Somit konnte neben dem reinen Datensatz auch auf die Erkenntnisse zur Auswahl und dem Training der Lern-Algorithmen zurückgegriffen werden. Bei den nachfolgenden Untersuchungen des Datenschemas werden im Rahmen dieser Arbeit die Ergebnisse dieses Papers zugrunde gelegt und durch eigene Experimente hinsichtlich der Gültigkeit bzw. Validität der bereits existierenden Resultate ergänzt.

2.6.3.2 Untersuchung der Datensätze

Im nachfolgenden Abschnitt wird in Kürze die Zusammensetzung und Konzeption des ausgewählten Datensatz erläutert. Der ausgewählte Datensatz CSE-CIC-IDS2018 untergliedert sich wiederum in vier einzelne Datensätze, die wiederum geordnet nach Tag und den an diesem Zeitpunkt ausgeführten Angriffen in mehreren Comma-Separated-Values (CSV) Dateien gespeichert wurden. Wie bereits in Kapitel 2.6.3.1 erwähnt, war neben der Erstellung des hier verwendeten Datensatzes eine Evaluierung verschiedener maschineller Lernmodelle und deren Fähigkeit zur Erkennung der verwendeten Angriffe Schwerpunkt der verwendeten Arbeit. Somit waren die CSV-Dateien bereits so weit aufbereitet, dass die Daten mit Überschriften (den sogenannten Features) und Label versehen waren. Im Rahmen des originalen Datasets wurde der normale von dem malignen Netzwerkverkehr durch die Label „Benign“ und der Art des Angriffs z.B. „Slowloris“ unterschieden. Diese Label wurden im späteren Verlauf dieser Arbeit in der Sektion zum Training des ML-Modells durch nummerische Werte, hier „0“ für „Benign“ und „1“ für die jeweilige Attacke, ersetzt. Die Erstellung der Datensätze fand von Seiten der Arbeitsgruppe mit einer selbstentwickelten Java-Anwendung statt, die aus den gesammelten Log- und Packet Capture (PCAP)-Dateien eine Auswahl an 80 unterschiedlichen Eigenschaften des Netzwerkverkehrs extrahierte und in die unterschiedlichen CSV-Dateien auslagerte.

Die unterschiedlichen Arten von Angriffsmustern wurden jeweils separiert während eines bestimmten Zeitfensters an einem Tage der Versuchswoche durchgeführt. Nach der Sammlung der Daten wurde mittels des RandomForestRegressor, einer Funktion aus der Python-Bibliothek Scikit-Learn, die Relevanz der Features für die Erkennung eines bestimmten Angriffsmusters bestimmt. Hieraus konnte auf Basis der Gewichtung der einzelnen Eigenschaften eine angriffsspezifische Teilmenge isoliert werden, die eindeutig für das Erkennen eines bestimmten Angriffsmusters war [23, S. 113]. Das Resultat dieser Untersuchung wird in der Tabelle im Anschluss zum Vergleich für die eigenen Berechnungen der Gewichtungen der einzelnen Eigenschaften („Feature Importance“) dargestellt:

Tabelle 19 Gewichtung der einzelnen Eigenschaften auf Basis des RandomForestRegressor (Quelle: Angelehnt an Tabelle aus [23, S. 113]

Label	**Feature**	**Gewicht**

DoS GoldenEye	Backward Packet Length Std	0.0479
	Flow IAT Min	0.0317
	Forward IAT Min	0.0257
	Flow IAT Mean	0.0214
DoS Slowloris	Flow Duration	0.0431
	Forward IAT Min	0.0378
	Backward IAT Mean	0.0300
	Forward IAT Mean	0.0265
DDoS LOIC	Backward Packet Length Std	0.1728
	Average Packet Size	0.0162
	Flow Duration	0.0137
	Flow IAT Std	0.0086

In den anschließenden Kapiteln werden die Analysen im Rahmen dieser Arbeit auf Basis der vorhandenen Datensätze aufgeführt und mit den bereits bestehenden Ergebnissen verglichen.

2.6.3.2.1 Analyse des GoldenEye-Datensatzes

Auf Basis der geringeren Größe dieses Datensatzes und der somit vermuteten schnelleren Ladezeiten begann die Analyse im Rahmen dieser Thesis mit den Werten für GoldenEye und Slowloris. Der Zeitraum für die DoS-Angriffe durch GoldenEye begann um 9:26 und endete um 10:09 Uhr am Donnerstag, den 15.02.2018. Für Slowloris wurde am selben Tag das Zeitfenster von 10:59 bis 11:40 Uhr ausgewählt. Der DDoS-Angriff unter Zuhilfenahme von LOIC fand am Dienstag, den 20.02.2018, von 10:12 bis 11:17 statt. [36] Auf die Eigenschaften und Auswertungen des LOIC- und des Slowloris-Datensatzes wird in den folgenden beiden Kapiteln eingegangen. Infolge des kurzen Abstandes der GoldenEye und Slowloris Angriffe war der protokollierte Netzwerkverkehr innerhalb einer CSV-Datei konkludiert und benötigte noch eine Trennung auf Basis der Label mit den Angriffsnamen. Bedingt durch die fehlenden Möglichkeiten im Bereich Data Science und der Datenauswertung von Seiten des .NET Frameworks und C# wurden die nachfolgenden Berechnungen der „Feature Importance“ mit Python-Skripten erstellt, die im Rahmen dieser Arbeit angefertigt wurden und im GitHub Repository unter dem Verzeichnis „ML.Proxy.FeatureImportance“ hinterlegt sind. Die hierbei verwendete Python Version war 3.9.5 [37] und die verwendeten Bibliotheken waren neben Numpy [38], Pandas [39] und Matplotlib [40] die Scikit-Learn Bibliothek [41] für die Verwendung des RandomForestRegressor. Zum Vergleich wurde mit der Bibliothek TensorFlow Decision Forests [42] ausgeführt in Jupyter Notebooks [43] dieselbe Analyse nochmals durchgeführt, um einen Vergleich zwischen den erhaltenen Werten zu erstellen. Die Verwendung der Notebooks war hierbei unumgänglich, da sich der Decision Forest nach mehreren Anläufen nicht innerhalb eines Skriptes hat ausführen lassen. Mittels der Scikit-Learn Bibliothek wurden drei Analysen der „Feature Importance“ für die einzelnen Eigenschaften

des Datensatzes durchgeführt. Die dargestellten Diagramme sind Erzeugnisse der oben aufgeführten Matplotlib Bibliothek und wurden während der Laufzeit des Skriptes generiert. Beginnend mit dem mitgelieferten Funktionsumfang des Random Forest Algorithmus, der die Gewichtung basierend auf dem sogenannten „Gini-Index“ berechnet.

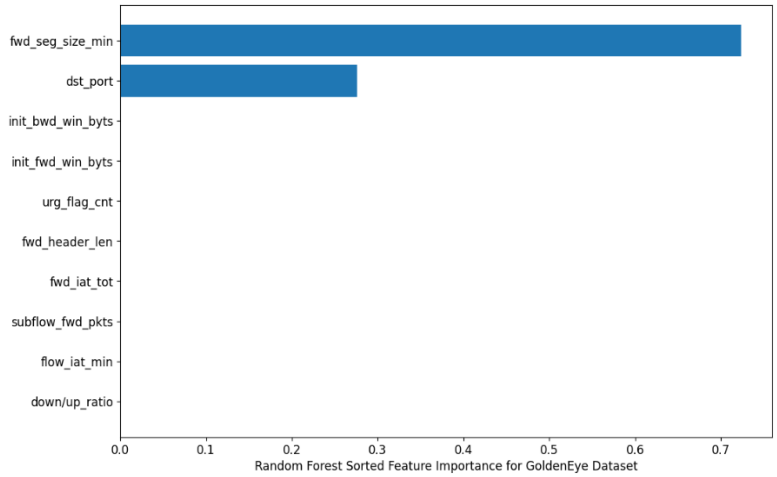

Abbildung 16 Auswahl der obersten zehn Eigenschaften auf Basis der „Feature Importance“ des RandomForestRegressor für das GoldenEye-Dataset (Quelle: Eigene Abbildung)

Wie bereits aus der Abbildung zu erkennen ist, befinden sich die Eigenschaften aus dem Paper nicht in den selbstberechneten Gewichtungen. Zum Vergleich wurde mit der TensorFlow Decision Forest Bibliothek eine ergänzende Evaluierung der Eigenschaften durchgeführt:

```
1   ("fwd_seg_size_min" (1; #51), 43.0),
2   ("init_fwd_win_byts" (1; #58), 39.0),
3   ("flow_pkts/s" (1; #33), 35.0),
4   ("flow_iat_mean" (1; #30), 31.0),
5   ("fwd_header_len" (1; #37), 21.0),
6   ("fwd_pkts/s" (1; #48), 21.0),
7   ("fwd_iat_max" (1; #38), 19.0),
8   ("flow_iat_max" (1; #29), 18.0),
9   ("timestamp" (4; #73), 12.0),
10  ("fwd_iat_tot" (1; #42), 11.0)
```

Abbildung 17 „Feature Importance“ für das GoldenEye-Dataset basierend auf den Berechnungen des TensorFlow Decision Forests (Quelle: Eigene Abbildung)

Auch hier sind Abweichungen zu den ausgewählten Features in der obigen Referenz zu erkennen. Einzig die Eigenschaft „flow_iat_mean“ in Zeile 4, die die „Interarrival Time“ zwischen dem Start von zwei „Flows“ angibt, stimmt mit dem vierten selektierten Kriterium des Originals überein. Hierauf folgte als nächstes die Auswertung mit der „Permutation Feature Importance“ [44], die auf Basis eines beispielhaft trainierten Modells die Eigenschaften des von dem Training-Datensatzes abgespaltenen Test-Datensatzes verwendet und durch zufälliges Einsetzen die Auswirkungen auf die Performanz des Modells analysiert.

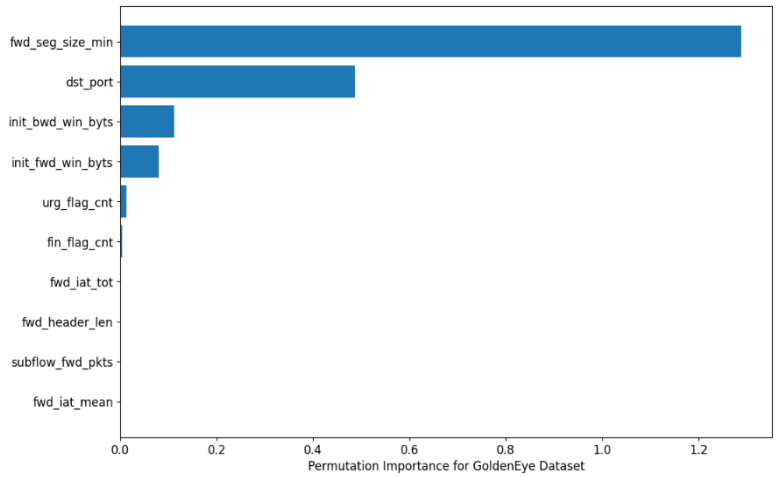

Abbildung 18 Die zehn am stärksten gewichteten Eigenschaften auf Basis der "Permutation Feature Importance" für das GoldenEye-Dataset (Quelle: Eigene Abbildung)

Auch in diesem Fall sind wie bei den vorherigen Analysen starke Abweichungen zu den ursprünglichen Werten aufgetreten. Durch die fehlenden Beschreibungen in den Aufzeichnungen zur Berechnung der „Feature Importance“ in der zugrundeliegenden Arbeit, lassen sich die festgestellten Unterschiede nur schwer konkretisieren. Die möglichen Variationen, die von Seiten der Bibliotheken geboten werden, die ein spezifisches Gewichten der Eigenschaften oder eine erhöhte Anzahl der verwendeten Bäume innerhalb des „Random Forest“ Modells ermöglichen, sind nicht im Paper enthalten. Somit sind Rückschlüsse auf Basis der eigenen Untersuchungen nur schwer durchführbar. Hierdurch ergeben sich weitere Aspekte zur Evaluation der Kriterien und Optionen, die eine Optimierung des Modells ermöglichen. Jedoch ist eine Verbesserung des Modells nicht Teil dieser Arbeit. Aus diesem Grund werden für das Training des später verwendeten Modells die selektierten Eigenschaften aus der Tabelle 19 verwendet. Hinsichtlich der Vollständigkeit werden in den anschließenden zwei Kapiteln die Gewichtung der Eigenschaften des Slowloris- und des LOIC-Angriffs dennoch berechnet.

2.6.3.2.2 Analyse des Slowloris-Datensatzes

Die in diesem Abschnitt verwendeten Ansätze und Analysen sind identisch zu den im Kapitel 2.6.3.2.1 und werden aus diesem Grund nicht nochmals in kompletter Ausführlichkeit erläutert. Nachfolgend werden nun die Ergebnisse der Berechnungen mit dem bereits vorgestellten Python-Skript und einem zusätzlichen Jupyter Notebook dargestellt.

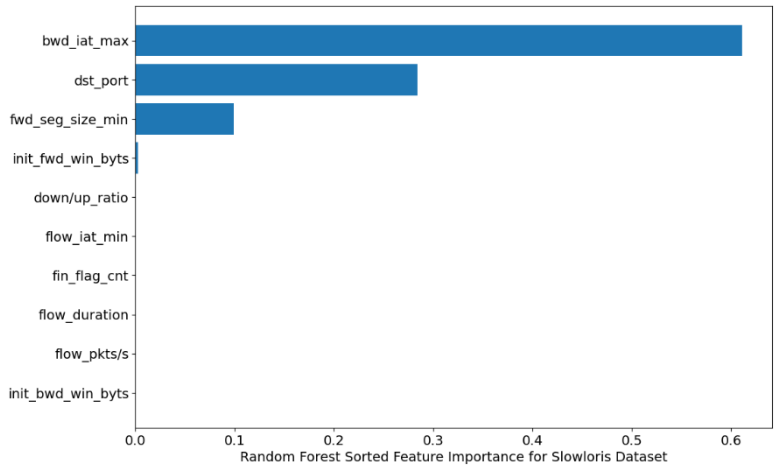

Abbildung 19 Auswahl der obersten zehn Eigenschaften auf Basis der „Feature Importance" des RandomForestRegressor für das Slowloris-Dataset (Quelle: Eigene Abbildung)

Wie im Fall des GoldenEye-Datensatzes sind in der Bewertung der Wichtigkeit der Eigenschaften für das Erkennen eines Slowloris-Angriffs unterschiedliche Ergebnisse im Vergleich zu der zugrundeliegenden Arbeit entstanden.

```
1   ("bwd_iat_max" (1; #8), 44.0),
2   ("bwd_iat_mean" (1; #9), 33.0),
3   ("bwd_iat_min" (1; #10), 32.0),
4   ("fwd_seg_size_min" (1; #51), 31.0),
5   ("bwd_iat_std" (1; #11), 25.0),
6   ("init_fwd_win_byts" (1; #58), 23.0),
7   ("flow_iat_std" (1; #32), 20.0),
8   ("active_max" (1; #1), 14.0),
9   ("active_mean" (1; #2), 14.0),
10  ("idle_std" (1; #56), 13.0)
```

Abbildung 20 „Feature Importance" für das Slowloris-Dataset basierend auf den Berechnungen des TensorFlow Decision Forests (Quelle: Eigene Abbildung)

Wie das Ergebnis der „Feature Importance" der TensorFlow Decision Forest Bibliothek zeigt, konnte weder die Scikit-learn Bibliothek noch die Implementierung von TensorFlow übereinstimmende Ergebnisse liefern.

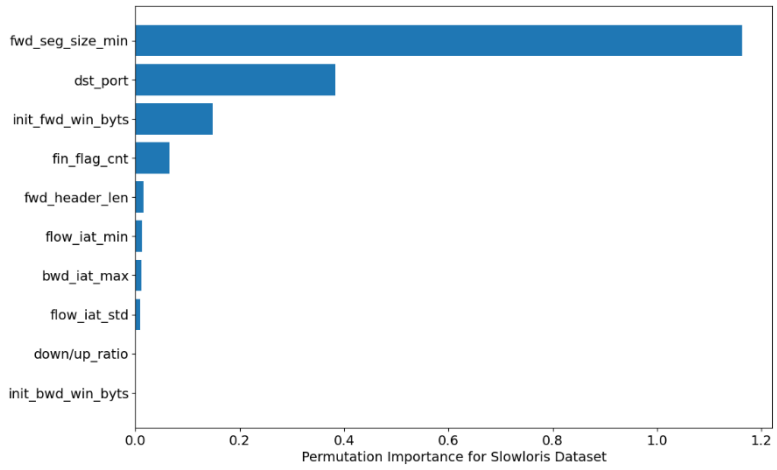

Abbildung 21 Die zehn am stärksten gewichteten Eigenschaften auf Basis der "Permutation Feature Importance" für das Slowloris-Dataset (Quelle: Eigene Abbildung)

Diese Differenzen setzen sich in der alternativen Analyse mit der „Permutation Feature Importance“ fort. Im Vergleich zu der Auswertung bei GoldenEye, gibt es hier keinerlei Übereinstimmungen mit dem Original. Wie in der Beurteilung des Ergebnisses aus dem vorherigen Kapitel, wird auch hier für das spätere Training eines Modells die vordefinierten Feature-Sets aus dem herangezogenen Paper verwendet.

2.6.3.2.3 Analyse des LOIC-Datensatzes

Abschließend erfolgte die Prüfung der „Feature Importance“ für das LOIC-Dataset. Jedoch waren auch hier die Ergebnisse vergleichbar mit den Feststellungen aus den vorherigen beiden Kapiteln. Weder die Scikit-learn Bibliothek noch die TensorFlow Decision Forest Bibliothek konnten dieselben Ergebnisse liefern.

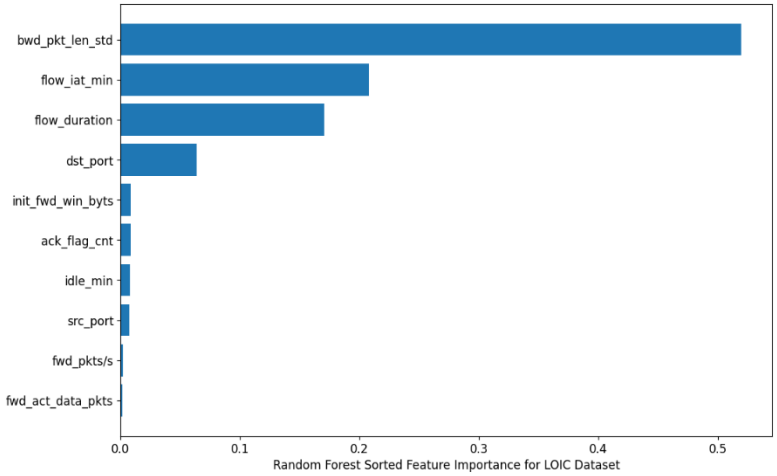

Abbildung 22 Auswahl der obersten zehn Eigenschaften auf Basis der „Feature Importance“ des RandomForestRegressor für das LOIC-Dataset (Quelle: Eigene Abbildung)

Es konnten zwar einzelne Übereinstimmungen wie z.B. aus Abbildung 22 ersichtlich die „bwd_pkt_len_std“, also die Standardabweichung der Größe von Paketen, die wieder vom angesteuerten Server zurückkommen und die „flow_duration“, als Dauer des einzelnen Flows, erzielt werden, jedoch sind dies nur zwei von vier Eigenschaften, die für eine valide Aussage somit nicht ausreichen.

```
("totlen_fwd_pkts" (1; #81), 52.0),
("dst_ip" (4; #24), 38.0),
("subflow_fwd_byts" (1; #74), 36.0),
("fwd_pkt_len_max" (1; #45), 28.0),
("fwd_seg_size_avg" (1; #52), 22.0),
("fwd_pkt_len_mean" (1; #46), 20.0),
("fwd_pkts/s" (1; #50), 18.0),
("flow_iat_max" (1; #30), 13.0),
("flow_pkts/s" (1; #35), 13.0),
("flow_duration" (1; #29), 12.0)
```

Abbildung 23 „Feature Importance“ für das LOIC-Dataset basierend auf den Berechnungen des TensorFlow Decision Forests (Quelle: Eigene Abbildung)

Keine Kongruenzen wurden von Seiten des TensorFlow Frameworks erzielt. Dieses konnte nach der Evaluierung des Datasets keine Übereinstimmung mit den gegebenen Feature-Sets erzielen.

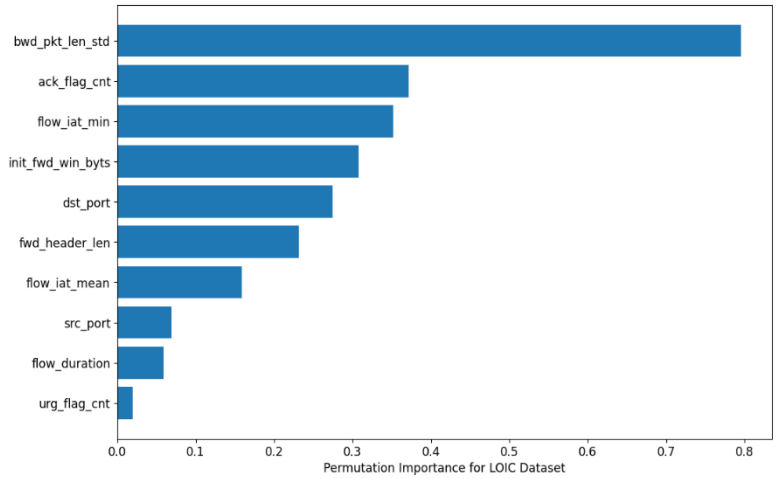

Abbildung 24 Die zehn am stärksten gewichteten Eigenschaften auf Basis der "Permutation Feature Importance" für das LOIC-Dataset (Quelle: Eigene Abbildung)

Wiederum vergleichbar war das Ergebnis der „Permutation Feature Importance“ Messung. Wie aus Abbildung 24 ersichtlich, konnte als markantestes Merkmal, wie auch bei der ersten Messung eine Konformität mit dem Kriterium „bwd_pkt_len_std“ erzielt werden. Trotz einer höheren Übereinstimmung mit den gegebenen Sets, werden auch in diesem Fall für das Modell die vorgegebenen Werte verwendet. Die Definition der nun ausgewählten Kriterien und das anschließende Training des Machine Learning Modells werden im folgenden Kapitel genauer behandelt.

2.6.3.3 Trainieren des ML.NET Modells mit den Daten

Die Gewichtung der Eigenschaften in den Datensätzen wurde somit abgeschlossen und die Auswahl der Features für das Training des Modells stand fest. Nachfolgend werden die selektierten Eigenschaften pro Angriff nochmals kurz aufgeführt und definiert. Die Definition basiert auf der der Legende in der Dokumentation zu den Datensätzen [23]. Hierbei sind die in Klammern gehaltenen Spaltenüberschriften das Produkt eines weiteren Python-Skriptes, welches die Überschriften vereinheitlicht und sie somit für die verschiedenen ML-Frameworks lesbar macht. Zum Beispiel ist es für die Bibliothek TensorFlow Decision Forests und auch ML.NET nicht möglich eine Überschrift bzw. Feature in den Trainingsdaten mit enthaltenen Leerzeichen zu erkennen und zu verarbeiten. Bei Testversuchen mit ML.NET war das Training zwar erfolgreich, jedoch hat die Validierung mit Testdaten nicht funktioniert. Aus diesem Grund wurde eine allgemeine Standardisierung eingeführt. Der Code für die Standardisierung und Selektion der Eigenschaften der nachfolgenden Features aus den ursprünglichen Datasets ist im GitHub Repository der hier vorliegenden Arbeit unter „ML.Proxy.Python.DataWrangler“ zu finden.

- DoS GoldenEye
    - Backward Packet Len Std („bwd_pkt_len_std“)
      Beschreibt die Standardabweichung der Länge eines Netzwerkpaketes in der Richtung von Server zu Client.
    - Flow IAT Min („flow_iat_min“)
      Beschreibt die minimale vergangene Zeit zwischen zwei Flows.
    - Forward IAT Min („fwd_iat_min“)
      Kennzeichnet die minimale vergangene Zeit zwischen dem Senden von zwei Paketen in der Richtung Client zu Server.
    - Flow IAT Mean („flow_iat_mean“)
      Beschreibt die durchschnittliche Zeit zwischen zwei Flows.
- DoS Slowloris
    - Flow Duration („flow_duration“)
      Kennzeichnet die Dauer eines Flows.
    - Forward IAT Min („fwd_iat_min“)
      Kennzeichnet die minimale vergangene Zeit zwischen dem Senden von zwei Paketen in der Richtung Client zu Server.
    - Backward IAT Mean („bwd_iat_mean“)
      Beschreibt die durchschnittliche vergangene Zeit zwischen dem Versenden von zwei Paketen in der Richtung Server zu Client.

    - Forward IAT Mean („fwd_iat_mean“)
      Beschreibt die durchschnittliche vergangene Zeit zwischen dem Versenden von zwei Paketen in der Richtung Client zu Server.
- DDoS LOIC
    - Backward Packet Len Std („bwd_pkt_len_std“)
      Bezeichnet die Standardabweichung der Größe eines Paketes in der Richtung von Server zu Client.
    - Avg Packet Size („pkt_size_avg“)
      Kennzeichnet die durchschnittliche Paketgröße.
    - Flow Duration („flow_duration“)
      Beschreibt die Dauer eines Flows.
    - Flow IAT Std („flow_iat_std“)
      Kennzeichnet die Standardabweichung der Zeit zwischen zwei Flows.

Diese Konstellation von Features mit den dazugehörigen Labels konnte somit in drei unterschiedliche CSV-Dateien ausgelagert werden, die jeweils einem Angriffsmuster entsprachen. Aufgrund der unterschiedlichen Eigenschaften jedes Angriffs und den damit verbundenen abweichenden „Feature Importance“-Kriterien fiel die Entscheidung im Rahmen dieser Arbeit anstatt einem Modell für die Erkennung aller drei Angriffe, drei unterschiedliche Modelle zu konzipieren, die jeweils spezifisch für eine Attacke trainiert werden. Ein weiterer Grund für diese Entscheidung war die prognostizierte steigende Genauigkeit für die Erkennung eines Angriffs. Das jeweilige Modell kann somit separiert von den zwei anderen spezifisch nach dem für ihn gedachten Angriffsmuster selektieren. Hiermit wird auch die Möglichkeit geboten unterschiedliche Services zur Erkennung von Angriffen auf die Cloud-Umgebung zu konzipieren. Eine mögliche Implementierung im ML.Proxy wäre eine Reihenschaltung der drei Modelle, die jede Anfrage in einer Art „Pipeline“ verarbeiten und auf Basis der Werte eine Einschätzung abgeben, ob es sich um einen zu blockenden oder einen validen Request handelt.

2.6.3.4 Implementierung eines Reverse Proxy mit Throttling-Mechanismus

2.6.3.5 Integration des Modells in den Proxy

3 Schluss

4 Literaturverzeichnis

[1] C. Duffy, *So you're one of 533 million in the Facebook leak. What now?* [Online]. Verfügbar unter: https://edition.cnn.com/2021/04/06/tech/facebook-data-leaked-what-to-do/index.html (Zugriff am: 8. April 2021).

[2] o.V., *Mehrere Schwachstellen in MS Exchange.* [Online]. Verfügbar unter: https://www.bsi.bund.de/SharedDocs/Cybersicherheitswarnungen/DE/2021/2021-197772-1132.pdf?__blob=publicationFile&v=4 (Zugriff am: 8. April 2021).

[3] o.V., *STP Informationstechnologie AG.* [Online]. Verfügbar unter: https://ka.stadtwiki.net/STP_Informationstechnologie_AG (Zugriff am: 6. Mai 2021).

[4] Nat Sakimura, John Bradley und Naveen Agarwal, *Proof Key for Code Exchange by OAuth Public Clients,* Request for Comments. RFC Editor. Verfügbar unter: https://rfc-editor.org/rfc/rfc7636.txt.

[5] o.V., *Was ist ein DDoS-Angriff?* [Online]. Verfügbar unter: https://www.cloudflare.com/de-de/learning/ddos/what-is-a-ddos-attack/ (Zugriff am: 28. Mai 2021).

[6] C. Eckert, *IT-Sicherheit, 10th Edition,* 10. Aufl. De Gruyter, 2018.

[7] A. Squicciarini, D. Oliveira und D. Lin, „Cloud Computing Essentials“ in *Cloud computing security: Foundations and challenges*, J. R. Vacca, Hg., Boca Raton, FL: CRC Press, Taylor & Francis Group, 2021, S. 3–11.

[8] W. Stallings, „An Overview of Cloud Computing“ in *Cloud computing security: Foundations and challenges*, J. R. Vacca, Hg., Boca Raton, FL: CRC Press, Taylor & Francis Group, 2021, S. 13–29.

[9] R. Mogull *et al., Security Guidance: For Critical Areas of Focus In Cloud Computing v4.0.* [Online]. Verfügbar unter: https://cloudsecurityalliance.org/artifacts/security-guidance-v4/ (Zugriff am: 6. Mai 2021).

[10] P. Wilmott, *Grundkurs Machine Learning,* 1. Aufl. Bonn: Rheinwerk Computing, 2020.

[11] P. Pandya und R. Rahmo, „Cloud Computing Architecture and Security Concepts“ in *Cloud computing security: Foundations and challenges*, J. R. Vacca, Hg., Boca Raton, FL: CRC Press, Taylor & Francis Group, 2021, S. 214–223.

[12] Bundesamt für Sicherheit in der Informationstechnik, *Cloud Computing Compliance Criteria Catalogue - C5:2020: Kriterienkatalog Cloud Computing.* [Online]. Verfügbar unter:

https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Informationen-und-Empfehlungen/Empfehlungen-nach-Angriffszielen/Cloud-Computing/Kriterienkatalog-C5/C5_AktuelleVersion/C5_AktuelleVersion_node.html (Zugriff am: 6. Mai 2021).

[13] *OWASP Zed Attack Proxy (ZAP)*. 2.10.0, 2021. [Online]. Verfügbar unter: https://www.zaproxy.org/

[14] *Security Code Scan - static code analyzer for .NET*. 5.1.0, 2021. [Online]. Verfügbar unter: https://security-code-scan.github.io/

[15] *Static Application Security Testing (SAST)*. [Online]. Verfügbar unter: https://docs.gitlab.com/ee/user/application_security/sast/index.html

[16] Congress Government, *H.R.200 – 117th Congress (2021-2022): National Intersection and Interchange Safety Construction Program Act of 2021*. [Online]. Verfügbar unter: https://www.congress.gov/bill/115th-congress/house-bill/4943/text (Zugriff am: 10. Mai 2021).

[17] o.V., *Qualitätsmanagement STP Cloud*. [Online]. Verfügbar unter: https://www.stp-online.de/qualitaetsmanagement/ (Zugriff am: 27. Mai 2021).

[18] *Oracle VirtualBox*. 6.1. Virtualisierungssoftware. Oracle, 2019. [Online]. Verfügbar unter: https://www.virtualbox.org/

[19] *Kali Linux*. 2021.1. OffSec Services Limited, 2013. [Online]. Verfügbar unter: https://www.kali.org/

[20] *Low Orbit Ion Cannon (LOIC)*. 2.0.0.4-1, 2009. [Online]. Verfügbar unter: https://github.com/NewEraCracker/LOIC

[21] *GoldenEye*. 1.0, 2012. [Online]. Verfügbar unter: https://github.com/jseidl/GoldenEye

[22] *slowloris.py - Simple slowloris in Python*. 0.2.3, 2015. [Online]. Verfügbar unter: https://github.com/gkbrk/slowloris

[23] I. Sharafaldin, A. Habibi Lashkari und A. A. Ghorbani, „Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization“ in *4th International Conference on Information Systems Security and Privacy*, Funchal, Madeira, Portugal, 1222018, S. 108–116, doi: 10.5220/0006639801080116.

[24] *NGINX*. 1.21.0. Reverse Proxy Webserver. F5 Inc., 2004. [Online]. Verfügbar unter: https://www.nginx.com/

[25] *.NET Core*. 5.0.7. Framework zur Erstellung von Applikationen mit C#. Microsoft, 2018. [Online]. Verfügbar unter: https://github.com/dotnet/core

[26] *ML.NET*. 1.5.5. Maschinelles Lern-Framework. Microsoft, 2019. [Online]. Verfügbar unter: https://github.com/dotnet/machinelearning

[27] *Yet Another Reverse Proxy*. 1.0.0-preview12. Reverse Proxy Framework für .NET-Applikationen. Microsoft, 2020. [Online]. Verfügbar unter: https://github.com/microsoft/reverse-proxy

[28] *Angular*. 12.1.1. Web-Framework für die Erstellung von SPA. Google, 2016. [Online]. Verfügbar unter: https://github.com/angular/angular

[29] *IdentityServer*. 5.2.1. OpenID Connect und OAuth2 Provider. Duende Software, 2020. [Online]. Verfügbar unter: https://github.com/DuendeSoftware/IdentityServer

[30] *Docker Compose*. 1.29.2. Tool für das Betreiben von Multi-Container Anwendungen mit Docker. Docker, 2013. [Online]. Verfügbar unter: https://github.com/docker/compose

[31] *Kubernetes*. 1.21.2. Container Orchestrierungssoftware. Google, 2014. [Online]. Verfügbar unter: https://github.com/kubernetes/kubernetes

[32] o.V., *Communications Security Establishment*. [Online]. Verfügbar unter: https://www.cse-cst.gc.ca/en (Zugriff am: 2. Juli 2021).

[33] Richard Lippmann, *1999 DARPA Intrusion Detection Evaluation*. [Online]. Verfügbar unter: https://www.ll.mit.edu/r-d/datasets/1999-darpa-intrusion-detection-evaluation-dataset (Zugriff am: 4. Juli 2021).

[34] D. Dua und C. Graff, *UCI Machine Learning Repository*. Verfügbar unter: http://archive.ics.uci.edu/ml.

[35] The Shmoo Group, *DEFCON 8, 10 und 11*. [Online]. Verfügbar unter: http://cctf.shmoo.com (Zugriff am: 4. Juli 2021).

[36] I. Sharafaldin, A. Habibi Lashkari und A. A. Ghorbani, *CSE-CIC-IDS2018 on AWS*. [Online]. Verfügbar unter: https://www.unb.ca/cic/datasets/ids-2018.html (Zugriff am: 2. Juli 2021).

[37] *Python*. 3.9.5. Python Software Foundation. [Online]. Verfügbar unter: https://www.python.org/

[38] Charles R. Harris *et al.,* „Array programming with NumPy“, *Nature*, Jg. 585, Nr. 7825, S. 357–362, 2020, doi: 10.1038/s41586-020-2649-2.

[39] Jeff Reback *et al., pandas-dev/pandas: Pandas 1.3.0*. Zenodo.

[40] J. D. Hunter, „Matplotlib: A 2D graphics environment“, *Computing in Science & Engineering*, Jg. 9, Nr. 3, S. 90–95, 2007, doi: 10.1109/MCSE.2007.55.

[41] F. Pedregosa *et al.,* „Scikit-learn: Machine Learning in Python“, *Journal of Machine Learning Research*, Jg. 12, S. 2825–2830, 2011.

[42] *TensorFlow Decision Forests*. 0.1.7. Maschinelles Lern-Framework für das Random Forest Modell. TensorFlow, 2021. [Online]. Verfügbar unter: https://github.com/tensorflow/decision-forests

[43] F. Loizides *et al., Hg., Jupyter Notebooks - a publishing format for reproducible computational workflows*, 2016.

[44] F. Pedregosa *et al., Permutation feature importance*. [Online]. Verfügbar unter: https://scikit-learn.org/stable/modules/permutation_importance.html#permutation-importance (Zugriff am: 4. Juli 2021).