Angewandte Computer- und Biowissenschaften

HOCHSCHULE MITTWEIDA
University of Applied Sciences

BACHELORARBEIT

Herr
Melvin Bayha

Entwicklung eines Konzepts für Sicherheitstests von Microsoft Azure Cloud-Infrastrukturen

2021

T · · Systems ·

M M S EXPERIENCE BEYOND DIGITAL

Fakultät **Angewandte Computer- und Biowissenschaften**

BACHELORARBEIT

Entwicklung eines Konzepts für Sicherheitstests von Microsoft Azure Cloud-Infrastrukturen

Autor:
Melvin Bayha

Studiengang:
IT-Sicherheit

Seminargruppe:
IF18wI1-B

Erstprüfer:
Prof. Dr.-Ing. Thomas Beierlein

Zweitprüfer:
Christoph Dehlan

Mittweida, 2021

Bibliografische Angaben

Bayha, Melvin: Entwicklung eines Konzepts für Sicherheitstests von Microsoft Azure Cloud-Infrastrukturen, 109 Seiten, 21 Abbildungen, 11 Tabellen, Hochschule Mittweida, University of Applied Sciences, Fakultät Angewandte Computer- und Biowissenschaften

Bachelorarbeit, 2021

Satz: LaTeX

Referat

Diese Arbeit befasst sich mit der Erstellung einer Vorgehensweise für das Durchführen von Sicherheitstests von Cloud-Infrastrukturen des Typs Microsoft Azure. Dafür werden anfangs der grundlegende Aufbau und die technische Realisierung der genannten Infrastruktur erarbeitet. Dies beinhaltet eine Feststellung der potenziellen Angriffsfläche nach außen. Ebenso geschieht eine Erfassung und Kontextualisierung des bereits bestehenden Wissens in Bezug auf Sicherheitstests von Azure-Cloud-Umgebungen. Im Detail umfasst dies die Generierung einer Übersicht über anerkannte und erprobte Vorgehensweisen, Best Practices, verfügbare und zu verwendende Tools sowie häufige Sicherheitslücken in Azure-Umgebungen. Darüber hinaus müssen offenkundige Grenzen innerhalb bestehender gesetzlicher Regularien sowie technischer Implementierungen für die Erstellung des Konzepts erfasst werden. Im Anschluss werden die erlangten Informationen genutzt, um vereinheitlicht eine Guideline zu entwerfen, nach welchem ein strukturierter Sicherheitstest einer Azure-Cloud-Infrastruktur durchgeführt werden kann. Letztendlich erfolgt eine vergleichende Diskussion der vorliegenden Feststellungen mit bestehenden anderen Vorgehensweisen, um diese Arbeit einzuordnen.

I. Inhaltsverzeichnis

II. Abbildungsverzeichnis

III. Tabellenverzeichnis

IV. Abkürzungsverzeichnis

AD	Active Directory
API	Application Programming Interface
ARP	Address Resolution Protocol
ASB	Azure Security Benchmark
ASC	Azure Security Center
AWS	Amazon Web Services
BDSG	Bundesdatenschutzgesetz
BSI	Bundesamt für Sicherheit in der Informationstechnik
C5	Cloud Computing Compliance Control Catalogue
CD	Continuos Delivery/Deployment
CI	Continuos Integration
CIS	Center for Internet Security
CPU	Central Processing Unit
CSA	Cloud Security Alliance
CSA CCM	CSA Cloud Control Matrix
CSP	Cloud Service Provider
D/DoS	(Distributed) Denial of Service
DSGVO	Datenschutzgrundverordnung
EICAR	European Institute for Computer Antivirus Research
FedRAMP	Federal Risk Authorization Management Program
GCP	Google Cloud Platform
GUI	Graphical User Interface
HIPAA	Health Insurance Portability and Accountability Act
HTML	Hypertext Markup Language
HTTPS	Hypertext Transfer Protocol Secure
IaaS	Infrastructure as a Service
IaC	Infrastructure as Code
IAM	Identity and Access Management
IoT	Internet of Things
IP	Internet Protocol

ISTQB ..... International Software Testing Qualifications Board

IT ..... Informationstechnologie

JEA ..... Just Enough Administration

JIT ..... Just-In-Time

JSON ..... JavaScript Object Notation

KI ..... Künstliche Intelligenz

MFA ..... Multi-Factor Authentication

NDA ..... Non-Disclosure Agreement

NIST ..... National Institute of Standards and Technology

OMS ..... Operations Management Suite

OWASP ..... Open Web Application Security Project

PaaS ..... Platform as a Service

PoC ..... Proof of Concept

RAM ..... Random Access Memory

RDP ..... Remote Desktop Protocol

REST ..... Representational State Transfer

SaaS ..... Software as a Service

SIEM ..... Security Information Event Management

SOAP ..... Simple Object Access Protocol

SOAR ..... Security Orchestration Automated Response

SP ..... Special Publication

SQL ..... Structured Query Language

SSH ..... Secure Shell

StGB ..... Strafgesetzbuch

TMG ..... Telemediengesetz

TTP ..... Tactics, Techniques and Procedures

US ..... United States

VHD ..... Virtual Hard Disk

VM ..... Virtual Machine

VMM ..... Virtual Machine Monitor

XSS ..... Cross-Site Scripting

1 Einleitung

1.1 Motivation

Infolge der stetig fortschreitenden Entwicklungen im Cloud-Bereich haben immer mehr Unternehmen begonnen, ihre Infrastruktur vollständig oder in Teilen nicht länger selbst zu betreuen. Stattdessen wird vermehrt auf die Nutzung der bezahlten Services verschiedener Cloud-Anbieter, darunter auch *Microsoft Azure*, gesetzt. Laut einer Statistik von *statista*[1] über die Nutzung von Cloud-Dienstleistungen in Deutschland (siehe Abbildung 1.1) nutzten im Jahr 2018 61 Prozent Dienste zur Datenspeicherung, dicht gefolgt von E-Mail-Services (48 Prozent) sowie Office-Anwendungen (34 Prozent) und Unternehmensdatenbanken (33 Prozent) [51]. Zwar gilt *AWS* (*Amazon Web Services*) als „die Nummer 1 in der Cloud“, gemessen an den Marktanteilen, jedoch ist *Microsoft* der Anbieter, der anstelle von *Amazon* einen Auftrag über zehn Milliarden US-Dollar (United States) erhalten hat, ein *Cloud Computing*-System für die US-Regierung aufzubauen [1].

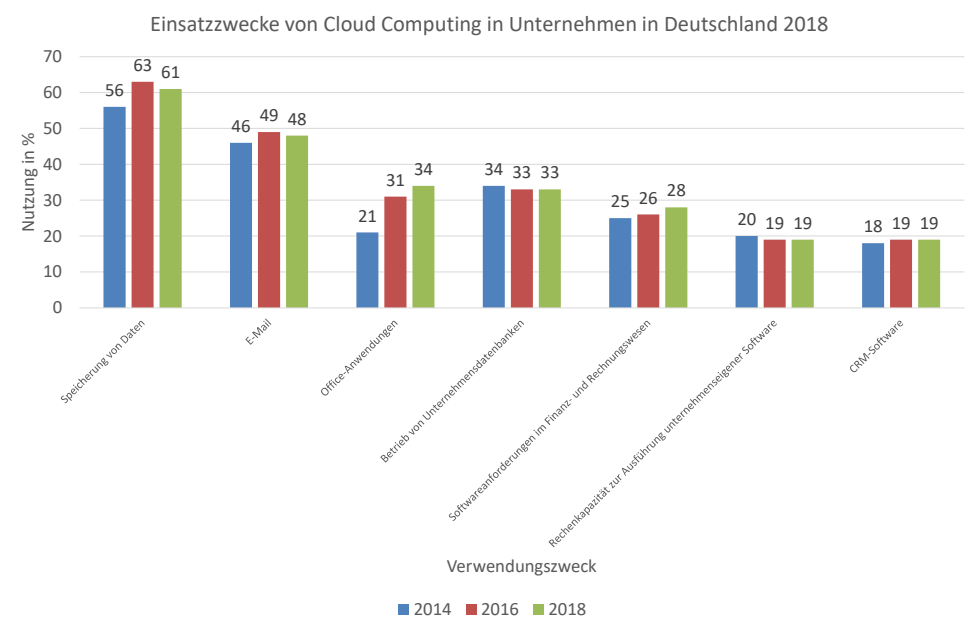

Abbildung 1.1: Diagramm über die Nutzung von Cloud Computing-Diensten verschiedener Kategorien in Deutschland [eigene Abbildung nach [51]]

[1] https://de.statista.com/statistik/daten/studie/381830/umfrage/einsatzzwecke-von-cloud-computing-in-unternehmen-in-deutschland/

Anhand dieser Zahlen wird deutlich, dass sensitive und wichtige Geschäftsdaten nicht länger nur auf den Geräten der jeweiligen Unternehmen und Organisationen, sondern auch auf einer Vielzahl von IT-Infrastrukturen (Informationstechnologie) weltweit verteilt sind. Dieser Umstand steigert die Attraktivität der Cloud als Angriffsziel, was beispielsweise durch den Angriff auf *DropBox* im Jahr 2014 sowie eine Studie über den Einsatz von so genannten *Honeypots* unterstützt wird. Somit muss die Cloud mindestens dasselbe Niveau an Sicherheit bieten können, wie eine gewöhnliche IT-Infrastruktur, um als in Betracht zu ziehende und ökonomische Alternative angesehen zu werden [5]. Aus diesem Grund ist es von zentraler Bedeutung, über Konzepte zu verfügen, wie solche Cloud-Infrastrukturen präventiv auf Sicherheitsprobleme und -lücken zu testen sind, um den bestmöglichen Schutz zu garantieren.

1.2 Zielstellung

Ziel dieser Arbeit ist die Konzeption einer Vorgehensweise für Sicherheitstests in *Microsoft Azure* Cloud-Infrastrukturen. Hierzu sollen existierende Anstrengungen in diesem Bereich erfasst und erläutert werden, bevor mithilfe des gewonnenen Wissens eine Empfehlung für deren Einsatz durchgeführt wird.

Zu Beginn werden wichtige Grundlagen und allgemeine Informationen zu den Thematiken Cloud, *Cloud Computing* und Sicherheitstests erfasst beziehungsweise definiert. Anschließend werden spezifische Informationen und Daten zu *Microsoft Azure* dargestellt. Begonnen wird die Erarbeitung möglicher Vorgehen für *Hardening Checks* und Penetrationstests in *Azure* mit einer Feststellung der Angriffsfläche sowie existierender Bedrohungen und Sicherheitsrisiken. Anschließend werden jeweilige Standards für diese beiden Arten von Sicherheitstests erfasst und untereinander bewertet. Abschließend werden gesetzliche, technische, fachliche sowie weitere Rahmenbedingungen erarbeitet.

Zentraler Inhalt dieser Vorgehensweisen sind Best Practices, Tools sowie insbesondere zu berücksichtigende Sicherheitslücken und -probleme. Basierend darauf soll das entworfene Konzept eine detaillierte Handlungsempfehlung für derartige Sicherheitstests von *Azure Cloud*-Infrastrukturen bieten. Abschließend wird diese neue Methodik gegenüber den bestehenden herangezogenen Werken diskutiert sowie bestehende Gemeinsamkeiten und Unterschiede herausgearbeitet.

2 Grundlagen

Die Cloud ist heutzutage im Alltag nahezu überall anzutreffen. Wird mit einem Smartphone ein Foto aufgenommen, steht dieses meist in direktem Kontakt mit der Cloud des jeweiligen Betriebssystem-Herstellers und lädt das wenige Sekunden alte Foto nahtlos hoch. Nicht nur Smartphones sind stets mit der Cloud verbunden: generell steht das gesamte *Internet of Things* (IoT) in einem ständigen Datenaustausch mit weit entfernten Servern [24].

Dabei ist die zugrundeliegende Funktionsweise von *Cloud Computing* kein neues Konzept. Bereits in den 1950er-Jahren stellten *Mainframes* den ersten Schritt in Richtung einer Cloud nach modernem Verständnis dar. Hierbei war die Funktionsweise jedoch noch eine völlig andere. Unternehmen und Universitäten stellten Großrechner zu Verfügung, welche von Mitarbeitern oder Studenten über eine gewisse Anzahl Terminals für deren Berechnungen genutzt werden konnten. Integraler Bestandteil dieses Ansatzes war das *Timesharing*: Rechenzeit musste reserviert werden, um das Angebot nutzen zu können [24].

Mit der Entwicklung der Virtualisierung wurde die Möglichkeit geschaffen, Recheninstanzen zu abstrahieren und rein virtuell zur Verfügung zu stellen. Mithilfe des Internets waren diese virtuellen Umgebungen auch online verfügbar, vorausgesetzt ein Zugriff auf das Internet bestand. Aus diesem Grund konnten derartige Modelle seit den 1990er-Jahren kommerziell der breiten Masse bereitgestellt werden [24].

Seither verzeichnet die digitale Welt eine explosionsartig ansteigende Entwicklung und Nutzung von Cloud-Technologien. Handelte es sich hierbei anfangs noch um Einzelleistungen, beispielsweise Plattformen für den einfachen Datenaustausch (*Google Spreadsheet, Google Docs*), entstanden zeitgleich große Server-Infrastrukturen z.B. von *Amazon*, welche über die Produktpalette der *Amazon Web Services* (AWS) heute von Firmen und anderen Interessenten dazu genutzt werden können, Software auszuführen [24]. Teil dieser großen Entwicklung im Bereich Cloud ist auch das Softwareunternehmen *Microsoft*. Zwar bekannt geworden durch das Betriebssystem *Windows*, bietet Microsoft heute mit seiner *Microsoft Azure* Cloud die bekannten Cloud-Dienstleistungen an (siehe Abschnitt 2.1, Unterabschnitt 2.1.3, Seite 9) [11].

Dieser Abschnitt thematisiert die Darstellung grundlegender Informationen über die Cloud. Hierbei sind insbesondere technische Definitionen, dahinterstehende Funktionalitäten und die angebotenen Dienste des *Cloud Computing* sowie die inhaltliche Definition einer *Cloud-Infrastruktur* von zentraler Bedeutung.

2.1 Cloud Computing

2.1.1 Definition

Im Folgenden ist die Definition des Begriffes *Cloud Computing* nach dem Internet-Dienstleister IONOS dargestellt (siehe Definition 2.1). Diese ist abgeleitet von einer Definition des **National Institute of Standards and Technology** (NIST). Die originale Definition des NIST ist in Definition 2.2 nochmals enthalten.

Definition 2.1: Unter dem Begriff *Cloud Computing* werden sämtliche Bereitstellungsangebote von Hardware und Software über das Internet gesammelt. Innerhalb dieses Rahmens ist dem Nutzer freigestellt, die angemietete Serviceleistung in Bezug auf Hardware und Software individuell zu skalieren, um seinen Bedarf zu decken. Der Umfang der Bereitstellung ist entsprechend flexibel, von der einfachen Anforderung weiteren Speicherplatzes (*Cloud Storage*) bis hin zu einer vollständigen Infrastruktur in der Cloud [24].

Definition 2.2: „*Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.*“[2]

Beide Definitionen beinhalten eine Reihe von Anforderungen, die eine Dienstleistung erfüllen muss, um demnach als *Cloud Computing* angesehen zu werden: Skalierbarkeit, Flexibilität, Allgegenwärtigkeit (Orts- und Zeitunabhängigkeit), Komfort (geringen Aufwand) sowie eine schnelle Bereitstellung und Freigabe verbunden mit minimalem Verwaltungsaufwand und Kontakt mit dem Dienstleister. Das NIST definiert diese Anforderungen im Detail als so genannte *Essenzielle Charakteristika*. Diese werden im Folgenden in Abschnitt 2.1.2 erfasst und verdeutlicht. Neben diesen gehören noch zwei weitere Bereiche zu dem Grundmodell der Cloud: *Service-* sowie *Deployment-Modelle* (siehe Abschnitte 2.1.3 und 2.1.4, Seiten 9 und 10) [14].

[2] MELL, Peter; GRANCE, Timothy: **The NIST Definition of Cloud Computing**, In: COMPUTER SECURITY DIVISION, INFORMATION TECHNOLOGY LABORATORY, NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (Hrsg.) *Special Publication 800-145*, 2011, Zugriff 10.06.2021, URL `https://csrc.nist.gov/publications/detail/sp/800-145/final`

2.1.2 Essenzielle Charakteristika

Gemäß der Definition des NIST existieren für das *Cloud Computing* fünf essenzielle Charakteristika. Diese werden in der folgenden Tabelle 2.1 nacheinander vorgestellt.

Tabelle 2.1: Essenzielle Charakteristika des Cloud Computing [eigene Tabelle nach [14]]

Eigenschaft	Beschreibung
On-Demand Self-Service	Dem Kunden ist die Möglichkeit gegeben, **nach Bedarf unilateral** (einseitig) Ressourcen (z.B. Serverzeit oder Speicherkapazitäten) für sich selbst bereitzustellen bzw. hinzu zu buchen. Hierfür ist **keine weitere Interaktion** mit dem jeweiligen Dienstleister erforderlich
Broad Network Access	Die Leistungen der Cloud sind **ausschließlich** über das Internet nutzbar. Hierbei werden **standardisierte Protokolle und Mechanismen** verwendet, welche die Nutzung über **gewöhnliche Endgeräte** (Notebooks, Computer, Smartphones) und somit für jeden erlauben
Resource Pooling	Die Rechenressourcen des Dienstleisters, beispielsweise Netzwerkbandbreite, Prozessorleistung oder Datenspeicherkapazität, werden **gebündelt**. Über ein Multi-Tenant-Modell (siehe Definition 2.3) werden diese jederzeit **dynamisch alloziert**. Entschieden wird über die Zuweisung **basierend auf dem Bedarf** des jeweiligen Kunden. Hierbei hat der Kunde **keine Kenntnis oder Kontrolle über den Herkunftsort** seiner Ressourcen, abgesehen von einer **abstrahierten Standortbestimmung** z.B. des Landes, der Region oder des Rechenzentrums

Fortsetzung von Tabelle 2.1

Rapid Elasticity	Die **Bereitstellung und Freigabe** von Ressourcen erfolgt **elastisch**, das heißt die Skalierung orientiert sich stets **angemessen an dem bestehenden Bedarf**. In manchen Fällen ist dieses Vorgehen **automatisierbar**, womit beim Kunden der Anschein entsteht, dass verfügbare Ressourcen im Grunde **unbegrenzt verfügbar** sind und eine **Zuweisung jederzeit** erfolgen kann
Measured Service	Zum Zweck der **effizienten Ressourcennutzung** werden Cloud-Systeme von dem jeweiligen Anbieter **Service-abhängig mit Messungen überwacht**. Dies ermöglicht die **Kontrolle, Überwachung, Optimierung und regelmäßige Meldung** der Auslastung genutzter Ressourcen und **fördert die Transparenz** für Anbieter und Kunde

Definition 2.3: *Multi-Tenancy* (*Mandantenfähigkeit*) beschreibt das Teilen einer einzelnen Instanz auf mehrere *Tenants* (*Mandanten*). Dies bedeutet beispielsweise bezogen auf Rechenressourcen, dass nicht jeder Mandant eigens ein weiteres Set Ressourcen (Prozessor, Random Access Memory (RAM), Datenspeicher etc.) erhält, sondern bestehende Ressourcen bedarfsgerecht aufgeteilt und neu zugewiesen werden [44].

2.1.3 Service-Modelle

Neben den *essenziellen Charakteristika* gehören zum Cloud-Modell des NIST eine Reihe unterschiedlicher *Service-Modelle*. Insgesamt existieren drei sich in ihrer Zielsetzung unterscheidende Services, die von einem *Cloud Computing*-Anbieter für gewöhnlich zur Verfügung gestellt werden. Hierbei handelt es sich um *Software as a Service* (SaaS), *Platform as a Service* (PaaS) und *Infrastructure as a Service* (IaaS) [14]. Im Folgenden werden die Definitionen dieser Modelle dargestellt.

Definition 2.4: Der Begriff *Software as a Service* beschreibt die Bereitstellung von Rechenkapazitäten einer Cloud für die Ausführung von angebotenen Softwarelösungen des Cloud-Anbieters. Hierbei hat der Kunde keinerlei Kontrolle über die zugrundeliegende Infrastruktur, in diesem Sinne die Hardware (Netzwerk, Speicherkapazitäten, Rechenleistung etc.) und Software (Betriebssystem, Treiber etc.). Darüber hinaus besteht auch bei der per SaaS bereitgestellten Software keinerlei Kontrolle von der Seite des Kunden, abgesehen von vereinzelten nutzerbezogenen Konfigurationseinstellungen [14].

Definition 2.5: *Platform as a Service* stellt im Gegensatz zu SaaS lediglich die Rechenkapazitäten der Cloud für die Ausführung von Softwarelösungen des Kunden zur Verfügung. Hierbei sind sowohl von dem Kunden selbst entworfene als auch erworbene Programme in das Service-Modell eingeschlossen, welche mithilfe von dem *Cloud Provider* unterstützten Programmiersprachen, Bibliotheken, Services o.Ä. erzeugt wurden. Dies schließt jedoch die Verwendung kompatibler Technologien aus anderen Quellen nicht aus. Der Kunde erhält volle Kontrolle über die Software und mögliche Konfigurationseinstellungen der Ausführungsumgebung. Die Kontrolle der Hard- und Software der Infrastruktur liegt weiterhin bei dem Anbieter [14].

Definition 2.6: Mit *Infrastructure as a Service* werden notwendige Systemressourcen wie Prozessorleistung, RAM, Speicherkapazität sowie Netzwerkkapazitäten und weitere Rechenressourcen bereitgestellt, mithilfe derer der Kunde beliebig Software und Betriebssysteme einsetzen kann. Die Kontrolle über die Hardware der Infrastruktur liegt zum größten Teil bei dem Provider. Ausnahmen stellen unter Umständen Firewalls oder ähnliche Netzwerkkomponenten und Speicherkapazitäten dar [14].

2.1.4 Deployment-Modelle

Abgeschlossen wird das allgemeine Cloud-Modell des NIST von den so genannten *Deployment-Modellen* für Cloud-Infrastrukturen. Diese beschreiben, wo die Infrastruktur sich physisch befindet, wer sie nutzt beziehungsweise nutzen darf und wie diese verwaltet wird. Insgesamt sieht das NIST vier *Deployment-Modelle* vor [14].

Private Cloud. Bei diesem *Deployment* wird die Infrastruktur einer einzigen Organisation für die alleinige Nutzung überlassen. Per Definition besteht eine Organisation hierbei aus mehreren Business Units. Besitz, Management, Operation und Hosting können von der Organisation selbst, einem Drittanbieter oder in Kombination dieser beiden Möglichkeiten vorgenommen werden. Dies lässt einen so genannten *On Premise*-Betrieb zu [14]. *On Premise* bezeichnet den Betrieb und die Verwaltung einer (erworbenen) Software oder Infrastruktur innerhalb der Jurisdiktion des eigenen Betriebsgeländes unter eigener Aufsicht und Verantwortung [8].

Community Cloud. Hier steht nicht die Einzelnutzung einer Infrastruktur, sondern die geteilte Nutzung dieser durch einen spezifisch ausgewählten Kreis von Kunden, basierend auf gemeinsamen Anliegen und Zielen. Hierbei handelt es sich beispielsweise um Compliance- oder Sicherheitsaspekte. Besitz, Management, Operation und Hosting dürfen zwischen den einzelnen Organisationen geteilt oder von einer allein übernommen werden. Auch eine dritte Partei ist wieder zur Übernahme der Verwaltungsaspekte befähigt, wodurch je nach Bedarf ein *On/Off Premise*-Betrieb ermöglicht wird [14].

Public Cloud. Wie der Name sagt, ist diese Art von *Deployment* öffentlich zugänglich und somit von jedem nutzbar. Das Management und Operation können dennoch an eine Organisation, beispielsweise eine Universität, Regierungsorganisation oder ein Unternehmen, fallen. Das Hosting findet *On Premise* bei dem *Cloud-Provider* statt [14].

Hybrid Cloud. Zuletzt existiert mit der *Hybrid Cloud* eine Mischung zweier oder mehr unterschiedlicher anderer *Deployment-Modelle*. Hierbei sind die Kombinationen *Public-Private*, *Public-Community* und *Private-Community* denkbar. Die beiden *Deployments* bleiben für sich einzelne Entitäten, teilen allerdings konforme oder herstellerspezifische Technologien, mithilfe derer die Adaption von beispielsweise Daten oder Anwendungen ermöglicht wird. Ein Beispiel hierfür ist das so genannte *Cloud Bursting* [14].

Definition 2.7: *Cloud Bursting* bezeichnet eine *Hybrid Cloud*-Konfiguration mit den beiden *Deployments Public-Private*. Dieses Setup dient der Handhabung von anormalen Ausschlägen bei der Auslastung. Beispielsweise würde bei einer vollständigen Auslastung einer *Private Cloud* eines Unternehmens weiterer Datenverkehr in eine *Public Cloud* transferiert, um eine ununterbrochene Erreichbarkeit aller Dienste zu gewährleisten. Mithilfe dieses Modells ist eine Kosteneinsparung möglich, da nicht für Ressourcen gezahlt werden muss, die nur in derartigen Fällen belastet würden. Stattdessen wird ein anderes *Deployment* herangezogen [43].

2.1.5 Das Shared Responsibility-Modell

Die Verantwortung für die Erstellung und Aufrechterhaltung einer sicheren Infrastruktur in *Azure* liegt nicht allein bei *Microsoft* als *Cloud Provider*, sondern wird mit dem Kunden in Abhängigkeit von den gewählten Service- und Deployment-Modellen variabel aufgeteilt. Dieses Vorgehen wird als *Shared Responsibility*-Modell bezeichnet (siehe Abbildung 2.1) [52, S. 4].

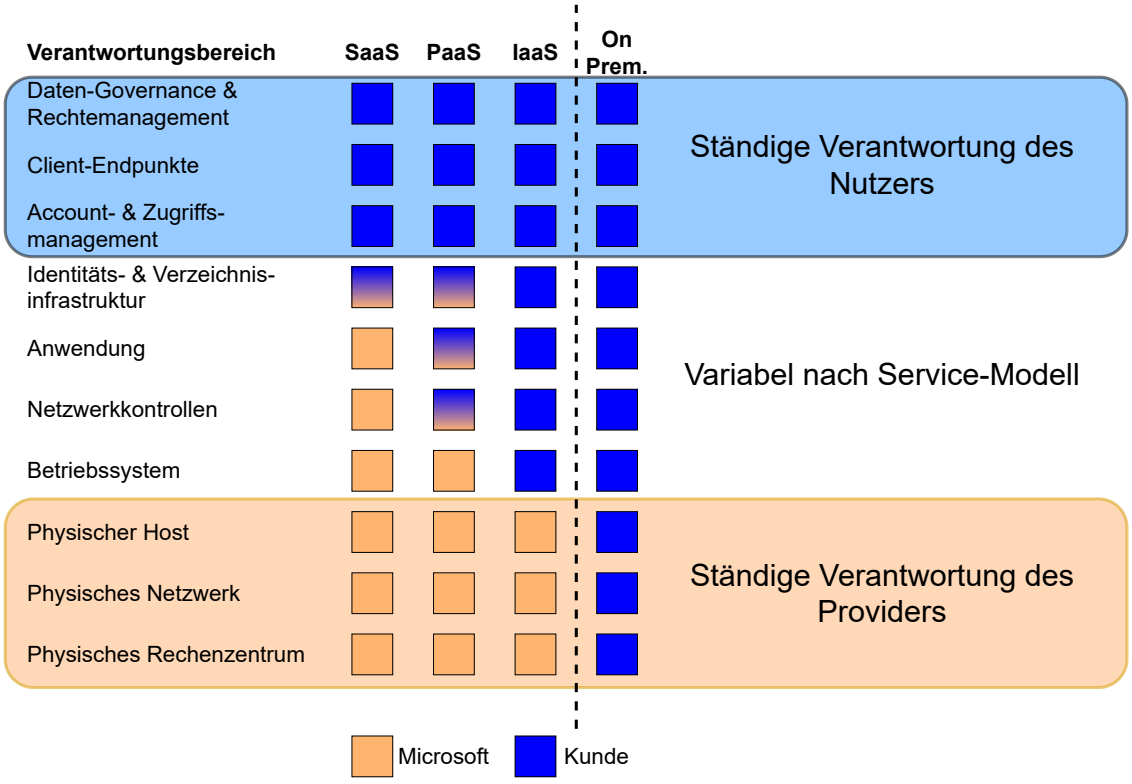

Abbildung 2.1: Darstellung der geteilten Verantwortung für verschiedene Sicherheitsaspekte zwischen Cloud-Nutzer und Cloud Provider [eigene Abbildung nach [52, S. 6]]

Eingeteilt wird das Modell in drei so genannte *Responsibility Zones*, übersetzt Verantwortungszonen. Diese unterteilen sämtliche Sicherheitsaspekte in **Ständige Verantwortung des Nutzers**, **Ständige Verantwortung des Providers** sowie **Variabel nach Service-Modell**. Der Nutzer ist stets verantwortlich für [52, S. 5-6]:

- Datenverwaltung und Rechtemanagement
- Endpoint Security
- Account- und Zugriffsmanagement

Microsoft auf der anderen Seite verwaltet immer folgende Aspekte [52, S.5-6]:

- Rechenzentrum
- Physische Netzwerke
- Physische Hosts/Server

Die Veranwortungsteilung verschiebt sich über nachstehende Aspekte [52, S. 5-6]:

- Identitätsmanagement und AD-Infrastruktur
- Applikationen
- Virtuelle Netzwerke
- Betriebssystem

Nachstehende Tabelle 2.2 veranschaulicht den Verantwortungsumfang eines Kunden in Abhängigkeit von dem gewählten Service-Modell.

Tabelle 2.2: Sicherheitskompetenzen des Kunden basierend auf dem Service-Modell [eigene Tabelle nach [52, S. 5-6]]

Service-Modell	**Kundenverantwortung**
On Premises	Jegliche Verantwortung in Kundenhand: physische Sicherheit, physische Netzwerksicherheit, Host-Sicherheit, Betriebssystem-Sicherheit, Anwendungssicherheit
IaaS	Kontrolle über Daten, Runtime und Anwendungen
PaaS	Kontrolle über Daten und Anwendungen
SaaS	Keine Verantwortung, abgesehen von wenigen Konfigurationsaspekten

2.2 Definition Cloud-Infrastruktur

Je nach Sichtweise ist eine Cloud-Infrastruktur unterschiedlich definiert. Dies hat den Grund, dass der Cloud-Provider und der Kunde abweichende Zielsetzungen verfolgen. Ziel des Providers ist der reibungslose Betrieb der Infrastruktur und dementsprechend die Bereitstellung der notwendigen Komponenten. Umgekehrt ist das Ziel des Kunden, sich aus diesem abstrakten Ressourcenpool gemäß den gesetzten eigenen Anforderungen bedarfsgerecht zu bedienen [50].

2.2.1 Definition aus Provider-Sicht

Aus Provider- beziehungsweise Betreibersicht bezeichnet die Cloud-Infrastruktur diejenigen Ressourcen, welche für den Betrieb einer Cloud allgemein erforderlich sind. Dies bedeutet vor allem Hardware, in diesem Kontext Server, Netzwerkkomponenten, Speicherkapazitäten sowie Rechenressourcen wie CPU (Central Processing Unit) und RAM. Darüber hinaus erfordert der Betrieb einer Cloud eine physische Infrastruktur in Form von Rechenzentren und Datenverbindungen zwischen diesen. Hier besteht die Möglichkeit, einen Drittanbieter anzuheuern oder diese Anforderungen selbst zu realisieren [50].

Neben dem hardwareseitigen Teil der Infrastruktur existiert ein ebenso komplexer softwareseitiger Teil. Dieser besteht aus Tools und Programmen für den täglichen Betrieb, die Verwaltung und Betreuung der Geschäftsprozesse und der Cloud-Infrastruktur selbst. Zusätzlich beinhaltet er Software, welche auf die Interaktion mit dem Kunden ausgelegt ist, beispielsweise um erbrachte Leistungen abzurechnen. Darüber hinaus existiert in den meisten Fällen direkt auf der Hardware eine Virtualisierungsschicht, auf der etwaige virtuelle Maschinen realisiert werden, beispielsweise für PaaS. Zuletzt besteht ein Bedarf an Software für das effiziente, tatsächlich elastische und leistungsfähige Management von *Workloads* (siehe Definition 2.8). Hier muss entsprechend gewährleistet sein, dass der Betrieb der Cloud beispielsweise im Wartungsfall dennoch reibungslos ablaufen kann [50].

Definition 2.8: Unter einer *Workload* wird in der IT die Arbeitsbelastung einer Komponente, beispielsweise einer CPU, oder einer Serviceeinheit (einem IT-System) verstanden. Im Rahmen des *Cloud Computing* bezeichnen *Workloads* einen plattformunabhängigen Service oder ein selbständig ausführendes Computerprogramm mit dem Ziel, eine festgelegte Aufgabe zu vollenden. Hierbei findet in der Cloud eine Abstraktion weg von der ausführenden Hardware statt. Das bedeutet, welches System den Service bearbeitet, ist unwichtig [10].

2.2.2 Definition aus Kundensicht

Aus Kundensicht ist die in Abschnitt 2.2.1 dargestellte Infrastruktur zu einem großen Teil unerheblich. Darüber hinaus spielt für die Nutzung eines Services (SaaS, PaaS oder IaaS) die Kenntnis der ausführenden Hardware keine weitere Rolle. Das Augenmerk des Kunden liegt auf der effizienten und flexiblen Nutzung von Ressourcen für seine Ziele, welche ihm in einem „normalen“ Serverraum nicht zur Verfügung stehen. Basierend auf dieser Einschätzung besteht die Cloud-Infrastruktur aus Kundensicht aus dem gewählten *Service*- und *Deployment*-Modell. Somit ergeben sich verschiedene Kombinationen aus diesen Modellen, die in ihrer Skalierung und Leistungsfähigkeit stets dem Zweck ihrer Nutzung angepasst werden können. Aus diesem Grund ist eine Cloud-Infrastruktur stets gemäß den Ansprüchen eines Kunden zu definieren [50].

Für ein entsprechend großes Unternehmen empfiehlt sich unter Umständen, das Rollenverständnis von Provider und Kunde zusammenzuführen. Gründe hierfür sind beispielsweise besonders strenge Compliance-Richtlinien. Eine exemplarische Kombination aus *Private Cloud* und IaaS würde resultierend daraus eine selbst verwaltete, skalierbare Zuordnung von Ressourcen ermöglichen, die zugleich nutzungsbasiert und auf Kostenstellen verteilt abgerechnet werden können. Die Anschaffung eigener physischer Ressourcen fiele in diesem Fall weg, es sei denn es handelt sich um ein *On Premise*-System [50].

2.3 Sicherheitstests in der Informationstechnik

Gemäß des NIST ist ein (Informations-)Sicherheitstest (engl. *information security assessment*) eine Feststellung darüber, wie vollständig und wirkungsvoll ein untersuchtes IT-System geltende Vorschriften und Richtlinien der Informationssicherheit erfüllt. Hierbei folgen derartige Tests einer festgelegten und standardisierten Struktur, um regelmäßige und wiederholbare Überprüfungen durchführen zu können sowie etwaige Risiken zu minimieren. Somit ist gegeben, dass die Testvorgehensweise einfach umzusetzen und simpel zu erlernen ist. Auch im Falle mehrerer Tester ist gewährleistet, dass ein reibungsloser Wechsel und in diesem Zuge eine Fortsetzung des Tests erfolgen kann [48, S. 2-1].

Für den Ablauf sollte eine Einteilung in Phasen in Erwägung gezogen werden. Die Anzahl ist hierbei nicht abschließend festgelegt, jedoch empfiehlt sich eine Vorgehensweise nach dem Muster **Vorbereitung - Durchführung - Nachbereitung**. Je nach Art des Tests sind diese Abschnitte unterschiedlich auszugestalten beziehungsweise zu ergänzen. Unterschieden wird dabei zwischen technischen Analyseverfahren.

Das Augenmerk liegt stets auf der Identifikation von Schwachstellen und Sicherheitslücken, allerdings liegt aufgrund der spezifischen Parameter der Verfahrensweisen der Fokus an anderer Stelle. Unterschieden wird in **Technische Überprüfung**, **Ziel- und Schwachstellen-Identifikation/-Analyse** und **Bestätigung von Ziel-Schwachstellen**. Diese sind in Abbildung 2.2 dargestellt und anschließend beschrieben [48, S. 2-1 bis 2-2].

Abbildung 2.2: Veranschaulichung der drei Verfahrenskategorien für Sicherheitstests [eigene Abbildung nach [48, S. 2-2 bis 2-3]]

Technische Überprüfung. Grundlage dieser Technik ist die Betrachtung und Bewertung von Dokumentationen, Log-Dateien, Regelwerken, Standards und Richtlinien, um die Sicherheit von IT-Systemen, Applikationen und Netzwerken einzuschätzen. Diese Art und Weise des Tests dient der umfassenden Untersuchung eines IT-Systems und sämtlicher Teilkomponenten, um Schwachstellen aufzudecken. Die Durchführung erfolgt in den meisten Fällen manuell [48, S. 2-2].

Ziel- und Schwachstellen-Identifikation/-Analyse. Hierbei sollen Systeme, Protokolle, Services und Ports sowie Schwachstellen in diesen identifiziert und analysiert werden. Mithilfe von Netzwerkscans, Port- und Service-Identifikation sowie Schwachstellenscans ist dieses Vorgehen als eine Mischung manueller und automatisierter Herangehensweisen anzusehen [48, S. 2-2 bis 2-3].

Bestätigung von Ziel-Schwachstellen. Auch diese Techniken verbinden manuelle und automatisierte Durchführung. Hierbei ist das Ziel die Verifikation von erfassten Schwachstellen in Systemen. Die Durchführung kann beispielsweise das Knacken von Passwörtern oder den Einsatz von *Exploits* beinhalten [48, S. 2-3].

Zuletzt unterscheidet sich bei Sicherheitstests der Ausgangs- beziehungsweise Startpunkt. Tests können entweder **offen** oder **verdeckt** sowie **extern** oder **intern** durchgeführt werden, was automatisch Einfluss auf den Umfang von Startinformationen hat, welche der Tester erhält. Ein externer Test beginnt mit den wenigsten Informationen und erfordert zu Beginn Aufklärungsarbeit, um einen Zugang zu dem gesetzten Testziel zu erwirken. Anschließend werden jegliche extern erreichbare IT-Systeme gemäß der zweiten und potenziell auch dritten beschriebenen Verfahrenskategorie untersucht. Interne Tests hingegen beginnen in der Position einer Person, welche bereits (offiziellen) Zugang zu dem Testziel besitzt. Hierbei findet vorrangig eine Untersuchung nach der ersten und zweiten Verfahrenskategorie statt. Die Unterscheidung zwischen offen und verdeckt bedeutet zudem eine Einteilung in Tests, bei welchen die IT-Mitarbeiter des Kunden vorab informiert beziehungsweise nicht informiert werden. Bei einem offenen Test besteht somit die Möglichkeit, durch fachkundige Anleitung den Test auf wichtige Aspekte eines Testziels zu fokussieren [48, S. 2-4 bis 2-5].

Im den folgenden Abschnitten 2.3.1 und 2.3.2 werden zwei Arten von Sicherheitstests, welche Bestandteil dieser Arbeit sind, gemäß der oben beschriebenen Merkmale vorgestellt und beschrieben. Das **Hardening** fällt in die Kategorie der **Technischen Überprüfung**. Die zweite in dieser Arbeit thematisierte Art des Sicherheitstests ist der **Penetrationstest**. Dieser fällt in die Kategorie **Ziel- und Schwachstellen-Identifikation/-Analyse** sowie unter Umständen **Bestätigung von Ziel-Schwachstellen**.

2.3.1 Hardening

Diese Art Test wird vom Bundesamt für Sicherheit in der Informationstechnik (BSI) auch als *technisches Sicherheitsaudit* bezeichnet und gehört der Verfahrenskategorie **Technische Überprüfung** an. Dieser Test wird intern und offen durchgeführt, in Zusammenarbeit mit den IT-Administratoren des Testziels. Das BSI veranschlagt neben dem *technischen Sicherheitsaudit* zwei weiterreichende Prüftiefen, den *nichtinvasiven* und *invasiven Schwachstellenscan*. Diese fallen in die Kategorie **Ziel- und Schwachstellen-Identifikation/-Analyse** sowie bei einem invasiven Scan in die **Bestätigung von Ziel-Schwachstellen**. Hierbei werden die Objekte, deren Härtung (Hardening) zu überprüfen ist, zusätzlich zur technischen (theoretischen) Überprüfung mit einem Schwachstellenscanner untersucht. Bei der nichtinvasiven Variante dieser Prüfung werden etwaige Ergebnisse nicht ausgenutzt. Umgekehrt wird bei der invasiven Variante versucht, mittels als geeignet eingestuften *Exploits*, die identifizierten Schwachstellen auszunutzen. Die Eignung eines *Exploits* ist abhängig von seiner potenziellen Schadwirkung [26, S. 17].

Definition 2.9: *Hardening*, vom BSI *technisches Sicherheitsaudit* genannt, bezeichnet die Praxis, die Angriffsfläche eines Computersystems, beispielsweise eines Servers, zu reduzieren. Hierbei werden eine Reihe von Software-Werkzeugen und simple Regeln für Konfigurationseinstellungen, so genannte *Best Practices*, eingesetzt, um potenzielle Sicherheitslücken und Angriffsvektoren zielgerichtet zu (er)schließen. Diese Vorgehensweise erstreckt sich über Nutzerrechte, Netzwerkports, Protokolle, Programme etc. Begleitet wird dieser Vorgang von einem Administrator des Kunden, welcher die zu testenden IT-Systeme präsentiert und mit dem Tester bespricht [26, S. 5, 17, 20].

Allgemeine Vorgehensweise für Hardening Checks gemäß BSI

Die Vorgehensweise des BSI sieht einen dreiteiligen Ansatz vor. Zu Beginn steht die *Einarbeitung der Prüfer*. Hierbei liegt das Augenmerk auf dem Studieren etwaiger bereitgestellter Dokumentationen. Vielmehr dient dieses Vorgehen der Identifikation potenzieller konzeptioneller Schwachstellen. Im Anschluss beginnt der eigentliche *Test des Prüfobjektes* oder *der Prüfobjekte*, abhängig vom Umfang des *Audits*. Diese Phase wird selbst in vier weitere Arbeitsschritte unterteilt, namentlich ein *Anfangsgespräch*, das *Einrichten der Arbeitsumgebung*, die *Praktische Prüfung* sowie ein *Abschlussgespräch*. Empfohlen wird eine Anpassung dieses Ablaufs, sollte es sich beispielsweise um einen mehrtägigen Test handeln. In einem solchen Fall wären auch *Zwischengespräche* denkbar. Abgeschlossen wird der dreistufige Aufbau mit einem schriftlichen Bericht, welche alle relevanten Informationen und Ergebnisse beinhaltet. Unter Umständen ist dieses Dokument aufgrund dessen Brisanz als *Vertraulich* zu kennzeichnen. Abbildung 2.3 (siehe Seite 20) veranschaulicht den allgemeinen Ablauf nach dem BSI. Zuvor erfolgt eine detaillierte Ausgestaltung dieses Vorgehens [26, S. 21-26].

1. **Einarbeitung der Prüfer**

    - Kennenlernen der IT-Systeme vor Ort
    - Sichten wesentlicher Dokumente (Dokumentationen, Arbeitsstandards etc.)

2. **Test des Prüfobjektes**

    - Anfangsgespräch

        – Gespräch mit IT-Verantwortlichen und Administratoren
        – Umfang des Prüfobjektes festlegen
        – Parameter des Prüfobjektes klären

    - Einrichten der Prüfumgebung

        – Überprüfung der Testvoraussetzungen
        – Zugang zu Prüfobjekten
        – Bereitschaft der Ansprechpartner

    - Praktische Prüfung

        – Modul I: Konzeptionelle Schwächen

            - Ableitung aus offenen Fragen zur Dokumentation

            - Erkennen organisatorischer Lücken

            - Aufbau & Infrastruktur des Testobjektes untersuchen

        – Modul II: Umsetzung Härtungsmaßnahmen

            - Überprüfung erforderlicher Härtungsmaßnahmen

            - *Offene Ports*

            - *Schnittstellen*

            - *Versionen und Patches*

            - *Anwendungszugang/Authentisierung*

            - *Sicherheit von Diensten & Richtlinien*

  - Modul III: Bekannte Schwachstellen

    - Einbeziehung Ergebnisse Modul II

    - Analyse von Patches und Softwareversionen

    - ggf. Nutzung von Schwachstellenscannern

  - Modul IV: Exploits

    - Schwachstellennachweis

    - Identifizieren geeigneter Exploits für bekannte Schwachstellen

    - Beachtung der Testparameter

- Abschlussgespräch

  - Informieren der Verantwortlichen über Verlauf und Ergebnisse
  - Festlegung der Anwesenden im Anfangsgespräch
  - Augenmerk auf kritischen Schwachstellen
  - Empfehlung: nicht alle Auswertungen vor Ort vornehmen, Minimierung der Testdauer

3. **Bericht**

- Bereitstellung von Prüfer für Auftraggeber und Verantwortliche
- Beinhaltet jegliche Ergebnisse
- Vertraulichkeiten berücksichtigen
- Technische Beschreibungen von Schwachstellen, Lösungsempfehlungen und Gruppierung nach Kritikalität

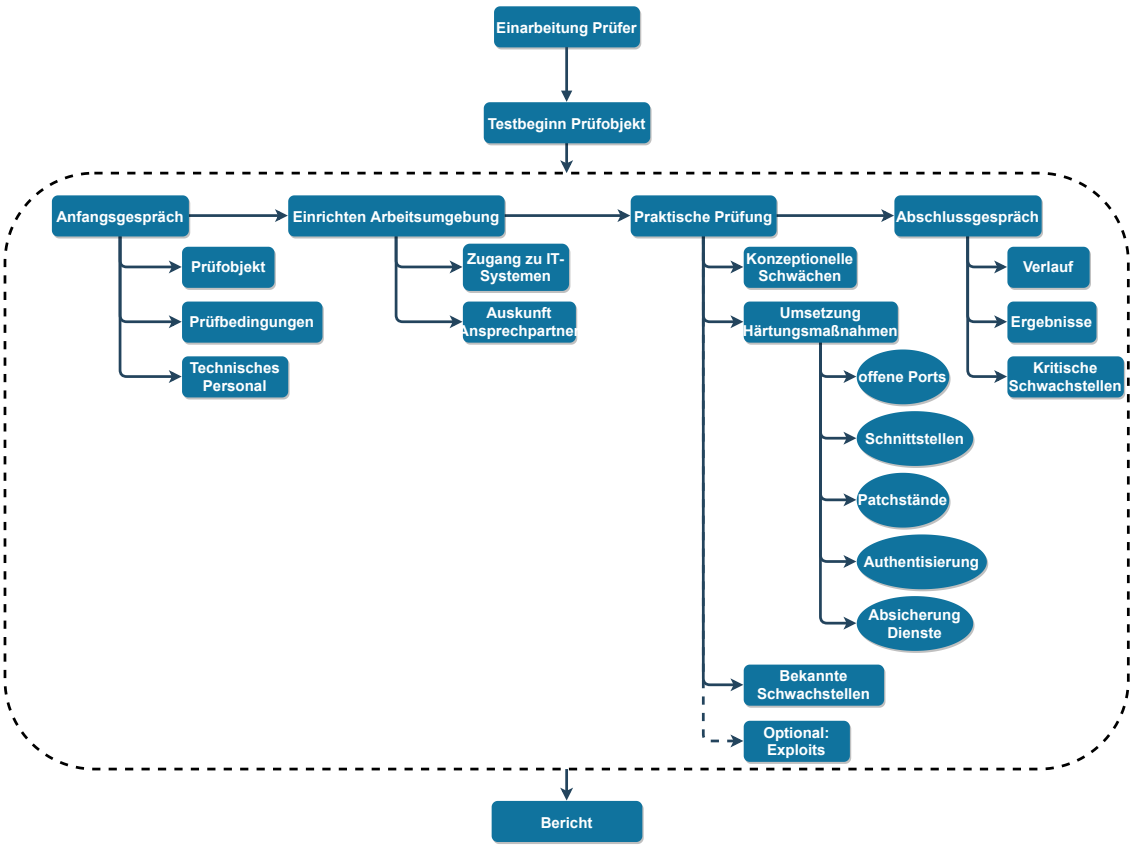

Abbildung 2.3: Allgemeine Vorgehensweise für ein Hardening (technisches Sicherheitsaudit) [eigene Abbildung nach [26, S. 21-26]]

Die Abbildung 2.3 stellt das schrittweise empfohlene Vorgehen des BSI für einen *Hardening Check* dar. Hierbei wird die Prüftiefe orientiert an den Zielen und Prüfbedingungen, welche beim Anfangsgespräch nochmals festgehalten werden, in der praktischen Prüfung entsprechend umgesetzt. Die Tiefe des bloßen *technischen Sicherheitsaudits* beinhaltet somit eine Suche nach konzeptionellen Schwächen, die Überprüfung der Umsetzung von Härtungsmaßnahmen sowie eine Feststellung von bekannten Schwachstellen basierend auf erfassten Patchständen, Maßnahmenkatalogen und weiteren Rahmenbedingungen. Für einen *nichtinvasiven Schwachstellenscan* würde die Untersuchung auf bekannte Schwachstellen und den Durchlauf eines Schwachstellenscanners erweitert. Zuletzt findet für die dritte Prüftiefe, den *invasiven Schwachstellenscan* eine Ergänzung des Vorgehens mit der optionalen Anwendung von geeigneten *Exploits* statt [26, S. 21-26].

2.3.2 Penetrationstest

In Kontrast zu einem *Vulnerability/Security Scan*, welcher automatisiert von einer Software nach einem Regelset über bekannte Schwachstellen ausgeführt wird, ist ein Penetrationstest eine hybride Aufgabe, welche neben dem (anfänglichen) Einsatz automatischer Tools die händische Arbeit des Testers erfordert [12]. Eine detailliertere Unter-

scheidung legt nahe, dass verschiedene Verfahrensweisen des Penetrationstests existieren. Diese nennen sich *White Box*-, *Grey Box*- und *Black Box*-Test. Im Grunde stellt diese farbliche Unterteilung anschaulich die Ausgangsbasis eines Tests in Bezug auf die Unterscheidungen offen/verdeckt und extern/intern dar. Vielmehr erlaubt die Einteilung in *White*, *Grey* und *Black* eine genaue Klassifizierung des Testrahmens, initialen Zugangs zu dem Testobjekt sowie des Umfangs der bereitgestellten Informationen wie beispielsweise Dokumentationen [49]. Dies wird in Abbildung 2.4 genauer dargestellt.

Ein Penetrationstest dient der Findung und potenziellen Ausnutzung von Schwachstellen und Sicherheitslücken in einem IT-System [12]. Folgende Definition 2.10 basiert auf einer Studie zu Penetrationstests des BSI.

Definition 2.10: Mithilfe eines *Penetrationstests* versucht ein *Ethical Hacker* (allg. IT-Sicherheitsexperte), die Sicherheit eines IT-Systems gegen Manipulations- und Einbruchsversuche zu bewerten. Hierzu werden gezielte Angriffe auf das ganzheitliche System oder einzelne Teile dessen unternommen. Ein Penetrationstest dient weiterhin der Emulation des potenziellen Verhaltens eines echten Angreifers, in dem reale Mittel und Werkzeuge eingesetzt werden [38, S. 5-6].

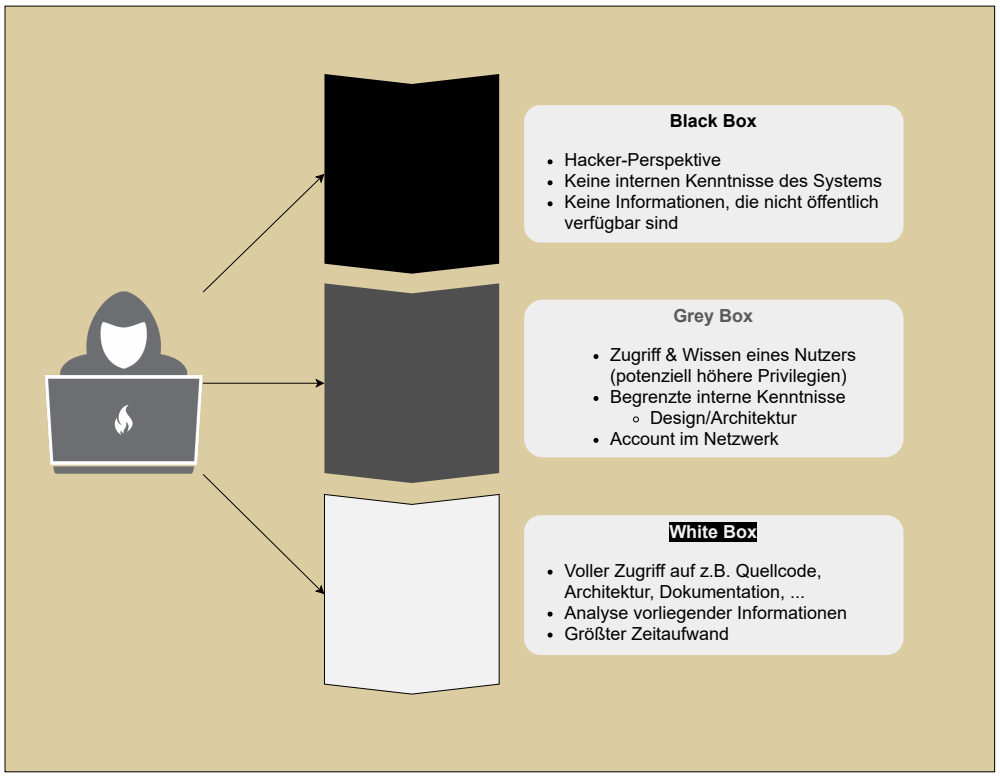

Abbildung 2.4: Black, Grey und White Box [eigene Abbildung nach [49]]

Ein *Black Box*-Test zeichnet sich durch den geringsten Gehalt an Startinformationen und der daraus resultierenden kürzesten Dauer aus. Kern des Tests ist eine dynamische Analyse der nach außen gerichteten Komponenten des Ziels, um festzustellen, inwiefern diese Sicherheitslücken aufweisen, welche von einem Angreifer ausgenutzt werden können. Hier muss ein Tester in der Lage sein, sich einen eigenen Überblick über das Ziel zu verschaffen, da beispielsweise bei einem Netzwerk keine weiteren Informationen als ein nach außen sichtbarer Ausgangspunkt bereitgestellt werden. Zudem erfordert diese Testart fundiertes Wissen im Umgang mit automatischen Scan-Werkzeugen sowie Methoden für das manuelle Testen bestimmter Protokolle, Services etc. Folglich hängt die Aussagekraft des Tests von der Fähigkeit des Testers ab, potenzielle Schwachstellen korrekt zu identifizieren beziehungsweise einen Weg zu finden, auf dem das Ziel kompromittiert werden könnte [49].

Dieser Test ist in den Verfahrenskategorien Ziel- und Schwachstellen-Identifikation/-Analyse sowie Bestätigung von Ziel-Schwachstellen einzuordnen. Zudem kann die Durchführung entweder offen oder verdeckt, jedoch stets extern erfolgen.

Währenddessen wird dem Tester bei einem *Grey Box*-Test eine breitere Fülle an Startinformationen angeboten. Dies erlaubt ein fokussiertes Vorgehen bei der Auswahl von Testfällen, beispielsweise nach der Größe des Risikos einzelner Systeme, wenn eine Dokumentation des Netzwerkaufbaus vorliegt. Darüber hinaus gestattet ein interner Zugriff über einen eigens angelegten Zugang dem Penetrationstester einen fundierten „Blick hinter die Kulissen“. Dieses Vorgehen wird als eine Verbindung von *Black* und *White Box*-Testing verstanden [49]. Somit ist diese Testart eine Kombination aus einer internen und externen Durchführung, welche jedoch offen erfolgt. Eingeordnet wird dieser Test in dieselben Verfahrenskategorien wie der *Black Box*-Test.

Auf der anderen Seite des Spektrums zum *Black Box*-Test erhält ein Tester bei einem *White Box*-Test jegliche verfügbaren Informationen, die für die Erfüllung des Testziels erforderlich sind. Hierbei liegt die Herausforderung in einer geordneten und umfangreichen Betrachtung des vorliegenden Wissens, um mögliche Probleme und Schwachstellen zu identifizieren. Dies ermöglicht eine realistische Simulation einer internen Bedrohung und des potenziell damit verbundenen Risikos. Durch den hohen Spielraum in der Informationsfülle ist ein *White Box*-Test in den meisten Fällen mit dem größten Zeitaufwand verbunden [49]. Dieser Test wird intern und offen durchgeführt. Zudem wird neben den Verfahrenskategorien der *Black Box*- und *Grey Box*-Tests auch die Technische Überprüfung einbezogen. Zwar ist der *White Box*-Test einem *Hardening* somit sehr ähnlich, jedoch ist bei einem *Hardening* stets ein Verantwortlicher des Kunden miteinzubeziehen, während ein *White Box*-Test von dem Tester allein durchgeführt werden kann.

Allgemeine Vorgehensweise für Penetrationstests gemäß BSI

Das BSI veranschlagt für einen Penetrationstest eine in fünf Phasen eingeteilte grundlegende Vorgehensweise. Zu Beginn steht die *Recherche nach Informationen über das Zielsystem*, gefolgt von einem *Scan der Zielsysteme auf angebotene Dienste*. Anschließend soll eine *System- und Anwendungserkennung* sowie eine *Recherche nach Sicherheitslücken/Schwachstellen* basierend auf den Ergebnissen der Service- und Systemuntersuchung erfolgen. Abgeschlossen wird dieses Vorgehen von einem *Ausnutzen der Schwachstellen* [38, S. 45-47]. Folgende Abbildung 2.5 verdeutlicht diese fünf Schritte sowie einzelne kategorisierte Aktionen.

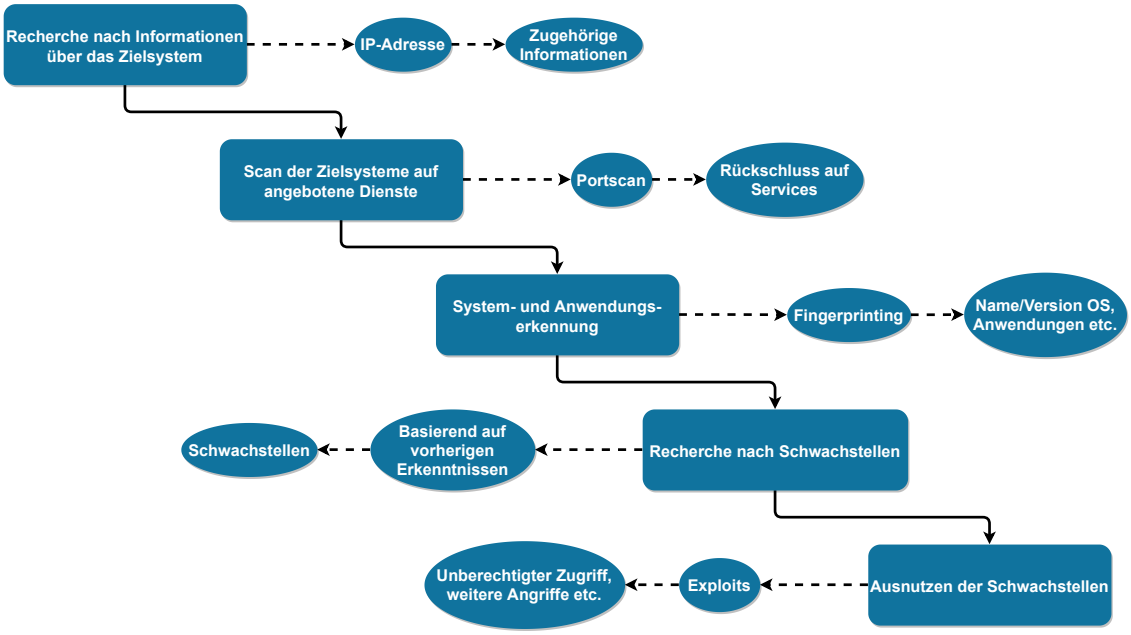

Abbildung 2.5: Allgemeine Vorgehensweise für einen Penetrationstest [eigene Abbildung nach [38, S. 45-47]]

2.3.3 Rahmenbedingungen

Für die Durchführung eines *Hardening Checks* beziehungsweise Penetrationstests in der Informationstechnologie existieren verschiedene Arten von Maßgaben, welche befolgt werden müssen. Hierbei handelt es sich um Gesetze, technische Vorgaben sowie weitere, variable Einschränkungen, denen Rechnung zu tragen ist. Die nachstehenden Abschnitte erläutern die unterschiedlichen Ausprägungen dieser Einschränkungen, um ein Gefühl für diese Grenzen zu vermitteln, welche währenddessen nicht überschritten werden sollten.

Geltendes Recht

Abhängig davon, in welchem Land sich der Organisationssitz eines Kunden befindet, werden andere Gesetze die Informationssicherheit und den Datenschutz betreffend veranschlagt. Wie diese Bestimmungen letzten Endes geartet sind, obliegt stets dem Gesetzgeber des jeweiligen Landes. Aus diesem Grund wird nachfolgend von den Gesetzen ausgegangen, welche Anwendung in Deutschland finden oder aber international anerkannt sind. Dies sind im Detail die **Datenschutzgrundverordnung** (DSGVO), das **Bundesdatenschutzgesetz** (BDSG), das **Telemediengesetz** (TMG) sowie aus dem **Strafgesetzbuch** (StGB) der Abschnitt **Verletzung des persönlichen Lebens- und Geheimnisbereichs** [15].

Zudem existiert für Organisationen, welche beispielsweise in den Vereinigten Staaten im Gesundheitssektor tätig sind, der *Health Insurance Portability and Accountability Act* (HIPAA) [30]. Aus den einzelnen gesetzlichen Vorgaben ergeben sich folgende zentrale Objektiven:

1: Gemäß DSGVO und BDSG, welche beide den Schutz personenbezogener Daten regulieren, die DSGVO in der EU und das BDSG auf deutscher Bundesebene, gilt, dass bei Kontakt mit derartigen Daten während eines *Hardening Checks* beziehungsweise eines Sicherheitstests im Allgemeinen die Weiterverbreitung und Verarbeitung zu unterlassen ist, da in diesem Fall kein berechtigtes Interesse besteht. Der Fokus des *Hardening* liegt auf der Überprüfung der Sicherheit eines *Azure*-Mandanten und personbezogene (sensible) Daten, welche (unabsichtlich) erfasst werden, sind ordnungsgemäß zu löschen und keinesfalls zu nutzen [25]. In dem Fall, dass solche Daten betroffen sind, sind die zuständigen Stellen in einer Organisation, z.B. der Datenschutzbeauftragte, zu informieren. Weiterhin sollten Versuche unternommen werden, diese Daten, wenn mit ihnen gearbeitet werden muss, zu anonymisieren. Besteht bei einer Überprüfung zudem der Kontakt zu als Verschlusssache eingestuften Informationen, sind die für Geheimschutz zuständigen Stellen einer Organisation in die Planung und Durchführung mit einzubeziehen [26, S. 14-15].

2: Prinzipiell steht *Hacking* und somit auch jegliche Art von Sicherheitstests in Deutschland unter Strafe (vgl. §202a ff. StGB). Dies gilt für den Fall, dass keine ausdrückliche Genehmigung des Betroffenen (in diesem Fall des Kunden) vorliegt, beispielsweise in Form eines geschlossenen Vertrages. Jedoch auch mit der Absicherung eines solchen Vertrages stehen vielfältige Vorkommnisse mit etwaigen elektronisch gespeicherten Daten des Kunden unter Strafe und sollten dementsprechend vermieden werden [37].

3: Laut §13, Abs. 7 TMG ist der Dienstanbieter eines Telemediendienstes in der Pflicht, insofern die technischen und wirtschaftlichen Möglichkeiten bestehen, seine Angebote technisch und organisatorisch zu sichern. Dies zielt insbesondere auf den Schutz personenbezogener Daten sowie auf die Verhinderung durch äußere Angriffe bedingter Störungen ab. Ebenso ist der unerlaubte Zugriff beispielsweise durch kryptographische Maßnahmen zu unterbinden. Hierbei ist stets der aktuelle Stand der Technik als Bewertungsmaßstab anzulegen [39].

Penetrationstest und *Hardening Check* fallen unter dieselben rechtlichen Beschränkungen. Dies bedeutet, dass die Rechtsgrundlage je nach Land, aus dem die Kundenorganisation kommt inklusive der Länder, in denen sich *Azure*-Knoten befinden, welche für die Realisierung etwaiger Services genutzt werden, variiert. Zudem greifen die gesetzlichen Vorgaben des Landes, in dem sich der Tester befindet [15].

Sollte eine gegebene IP-Adresse nicht dem Kunden, sondern einer anderen Organisation, beispielsweise einem Staat gehören, besteht unter Umständen das Potenzial für einen diplomatischen Zwischenfall, sollten die Rechte dieses verletzt werden [3, S. 3].

Technische und fachliche Vorgaben

Hardening Check: Der *Hardening Check* einer IT-Infrastruktur sollte laut dem BSI stets von Personen durchgeführt werden, welche hierfür fachlich qualifiziert sind, jedoch dabei nicht gleichzeitig mitverantwortlich für den zu überprüfenden Systemverbund sind oder bei dessen Konzeption und Aufbau mitgewirkt haben. Hiermit ergibt sich eine Vorbeugung von Interessenkonflikten oder Betriebsblindheit, wodurch eine objektive und umfassende Evaluation garantiert wird. Bevor jedoch ein *Hardening Check* durch einen externen Prüfer vorgenommen wird, sollte nach Möglichkeit bereits eine interne Qualitätssicherung stattgefunden haben, da diese nicht durch eine externe Überprüfung ersetzt werden kann [26, S. 9-10].

Neben den allgemeinen Anforderungen an die Durchführung eines *Hardening Checks* besteht eine Reihe technischer und fachlicher Anforderungen an den Prüfer selbst. Diese können entweder anhand erworbener Zertifikate überprüft oder aber anhand der Gegebenheiten des Prüfobjektes individuell erfasst werden. Beispielsweise ist ein erfahrener Tester von Web-Anwendungen nicht automatisch für die Überprüfung einer komplexen Infrastruktur geeignet. Die vom BSI veranschlagten Kenntnisse umfassen folgende Themenbereiche [26, S. 11-13]:

- Systemadministration
- Netzwerkprotokolle
- Gängige Programmiersprachen
- Bekannte IT-Sicherheitstechnologien (IDPS, Firewalls, Gateways, ...)
- Anwendungssysteme
- Netzwerkkomponenten

Speziell auf *Azure*-Infrastrukturen bezogen besteht darüber hinaus der Bedarf nach Kenntnissen im Umgang mit einer Vielzahl von *Azure*-Services, insbesondere mit sicherheitsbezogenen Einstellungen und Konfigurationsmöglichkeiten. Hier sind beispielsweise das *Azure Security Center*, *Azure Sentinel* sowie das *Azure AD* zu nennen. Neben Kenntnissen technischer Gegebenheiten in *Azure* ist dementsprechend auch ein Wissen über Vorgehensweisen und zu überprüfende Faktoren im *Azure*-Umfeld (siehe Kapitel 4, Abschnitt 4.2, Seite 50) förderlich. Letztlich kann auch der Umgang mit bekannten Werkzeugen, beispielsweise Schwachstellenscannern wie *Nessus*, vorausgesetzt werden.

Penetrationstest: Ein *Azure*-Knoten (Server eines Rechenzentrums) beherbergt aufgrund der Virtualisierung als Grundlage für *Cloud Computing* mehrere Services beziehungsweise Instanzen verschiedener Organisationen, welche *Azure* nutzen. Aus diesem Grund können IP-Adressen sehr wohl für einen einzelnen Kunden genutzt werden, jedoch auch zwischen verschiedenen geteilt sein. Eine Untersuchung, beispielsweise ein *Portscan* gegen diese Art IP ist im deutschen Rechtssystem nicht klar geregelt, könnte jedoch nach § 202c StGB als Vorbereitung für einen Angriff ausgelegt werden. Zudem verstoßen derartige Untersuchungen einer falschen IP gegen die *Rules of Engagement* für Penetrationstests der Firma *Microsoft* [3, S. 3-4] [35].

Zusätzlich besteht eine Limitation des Testrahmens, welcher das *Azure Fabric* und somit jegliche physische Infrastruktur von *Microsoft* sowie den *Hypervisor* als Virtualisierungsebene kategorisch ausschließt. Jegliche Untersuchungen in dieser Richtung bewegen sich außerhalb des Rahmens, der den Kunden eines Cloud-Penetrationstest betrifft [3, S. xxii-xxiii]. *Microsoft* verfügt aus diesem Grund über eine Reihe so genannter *Rules of Engagement*, übersetzt Einsatzregeln, welche technische und fachliche Rahmenbedingungen bieten. Aus diesen ergeben sich folgende Limitationen [35]:

- Kein Zugriff auf Eigentum (Ressourcen) anderer Kunden
- Kein Zugriff auf Daten, welche nicht vollständig dem Kunden des Tests gehören
- Keine (D)DoS-Tests (*(Distributed) Denial of Service*)
- Keine netzwerkintensiven Fuzzingvorgänge für Ressourcen außerhalb der virtuellen Computer des Testkunden
- Keine automatisierten Tests mit einem hohen Datenverkehr
- Kein absichtlicher Zugriff auf fremde Daten
- Keine Überschreitung von Kompetenzen bei *Proof of Concepts* (PoC)
- Keine Nutzung von Diensten, welche gegen die Richtlinien von *Microsoft* verstoßen
- Kein *Phishing* gegen *Microsoft*-Mitarbeiter

Neben diesen Beschränkungen präsentiert *Microsoft* ebenfalls Empfehlungen für die Durchführung eines Penetrationstests in *Azure*. Diese sollen dazu dienen, Testfälle festzulegen, welche im Rahmen der Bestimmungen dennoch durchgeführt werden, auch wenn für diese ein generelles Verbot besteht [35]:

- Erstellung von mehreren Testmandanten, um mandantenübergreifenden Datenzugriff nachzuweisen
- *Fuzzing*, *Portscanning*, Schwachstellenscans auf den virtuellen Computern des Testkunden
- Auslastungstests auf den Anwendungen des Testkunden, gemäß dem Datenverkehr, welcher im ordnungsgemäßen Betrieb anfallen könnte
- Durchführung von Tests mittels einer EICAR-Datei zur Reaktion von Sicherheitsüberwachung und -erkennung (*European Institute for Computer Antivirus Research*)
- Versuche, aus Containern oder Webseiten auszubrechen[3]
- Festlegung von Richtlinien in Zusammenhang mit der Verwaltung von Mobilgeräten bei *Microsoft Intune*, zum Testen der Durchsetzung

[3] Jeglicher Erfolg ist an *Microsoft* zu melden und der Test ist zu beenden

Weitere Grenzen

Hardening Check: Neben rechtlichen, technischen und fachlichen Bedingungen existieren weitere Vorgaben, welche entsprechend respektiert und eingehalten werden müssen. Dies ist zum Einen der geschlossene Vertrag mit dem Kunden. Dieser sichert die Vereinbarung über einen Sicherheitstest, in diesem Fall den *Hardening Check*, rechtlich ab und trägt dazu bei, vielfältige in Abschnitt Geltendes Recht (siehe Seite 24) beschriebene Konsequenzen zu vermeiden. Sollten Dienste nicht dem Kunden gehören, sondern bei einem Hoster ausgelagert sein, ist unter Umständen auch dieser, sofern nicht anders bekannt, in den Vertrag mit einzubeziehen. Im Fall von *Azure* trifft dies nicht zu, da *Microsoft* inzwischen nicht mehr voraussetzt, über Penetrationstests oder andere Überprüfungen bei *Azure*-Ressourcen informiert zu werden [26, S. 14] [34]. Allgemein definiert der Vertrag Kenngrößen wie Dauer, Prüfobjekte und Prüftiefe eines *Hardening Checks*. Dies sichert den Prüfer dagegen, dass dieser bei während der Überprüfung zufällig aufgetretenen Fehlern zur Verantwortung gezogen wird. Zudem ist festgelegt, welche Kosten anfallen und ob neben der reinen Prüfung der Zielsysteme weitere Dienstleistungen, wie beispielsweise eine Präsentation der Ergebnisse, erfolgen soll. Darüber hinaus enthält der Vertrag Informationen bezüglich Haftbarkeit und Verschwiegenheit des Auftragnehmers sowie einen Hinweis darauf, dass die ermittelten Ergebnisse ausschließlich zum Zeitpunkt des Tests Gültigkeit besitzen, wobei aufgrund der Rahmendaten keine Gewährleistung besteht, dass alle Risiken identifiziert werden konnten [26, S. 14].

Neben einem Vertrag ist ein Non-Disclosure Agreement (NDA) empfehlenswert, da somit klar geregelt ist, welche Informationen der Überprüfung an Dritte kommuniziert werden dürfen. Beispielsweise besteht die Möglichkeit, den Bericht eines *Hardening Checks* exemplarisch zu veröffentlichen oder anderen Parteien zugänglich zu machen, um gegenüber bestimmter Ergebnisse zu sensibilisieren oder ein Arbeitsbeispiel zu präsentieren. Jegliche sensible (personenbezogene) Daten müssen dennoch anonymisiert werden [26, S. 14].

Weiterhin ist die Speicherzeit von Daten eine vitale Maßgabe. Während eines *Hardening Checks* werden Daten erhoben und gespeichert, welche für die Erstellung eines Berichtes beziehungsweise die Erlangung der Ergebnisse genutzt werden. Hierbei ist zu regeln, wie lange und auf welche Art und Weise diese Daten vorzuhalten sowie ob gegebenenfalls Nachweise darüber zu erbringen sind [26, S. 14].

Durch einen *Hardening Check* betroffene Personen, beispielsweise Mitarbeiter oder Kunden, welche unter Umständen in ihrer Arbeit beeinträchtigt werden, sind über die Überprüfung zu benachrichtigen. Bei der Planung sollte zwar darauf geachtet werden, dass möglichst wenige Beeinträchtigungen entstehen, jedoch ist eine Benachrichtigung potenziell betroffener Personenkreise im Voraus ratsam [26, S. 15].

Zuletzt ist es empfehlenswert, darauf zu achten, dass während eines *Hardening Checks* keine Veränderungen an den zu untersuchenden Systemen vorgenommen werden, beispielsweise durch Wartungsarbeiten. Dies hängt damit zusammen, dass eine an diesen Systemen durchgeführte Überprüfung dadurch einerseits behindert werden kann und andererseits ihre Aussagekraft verliert, da die Bedingungen der Prüfung am Ende nicht dieselben sind, mit denen begonnen wurde [26, S. 15].

Penetrationstest: Über die rechtlichen, technischen und fachlichen Bedingungen hinaus bestehen verschiedene weitere Grenzen, welche bei einem Penetrationstest in *Azure* beziehungsweise der Cloud berücksichtigt werden sollten. Hierbei handelt es sich beispielsweise um die Limitationen bei der Entdeckung von Schwachstellen und Sicherheitsrisiken, wenn ausschließlich das *Cloud-Deployment* eines Kunden getestet wird. Dies hat den Grund, dass *Azure* von vornherein ein höheres Sicherheitsniveau aufweist. Der Grund hierfür liegt darin, dass *Microsoft* als der *Provider* ebenfalls gesetzliche Regulationen und industrielle Standards erfüllen muss, um konkurrenzfähig zu sein und seine Kunden zu halten. Werden beispielsweise VM-Images eingesetzt, welche von *Microsoft* erhältlich sind, weisen diese direkt nach dem *Deployment* in *Azure* hohe Sicherheitsmaßnahmen auf, beispielsweise einen einzigen Administrator sowie eine streng konfigurierte Firewall [3, S. 2].

Aus diesem Grund könnte ein hybrider Ansatz empfohlen werden, bei dem explizit darauf bestanden wird, in Beziehung stehende *On Premise*-Systeme des Kunden mit in den Test einzubeziehen. Auf diesem Wege ist davon auszugehen, dass die Anzahl der *Findings* sich signifikant erhöht, da beispielsweise Passwörter für lokale und *Azure*-Administrationskonten gedoppelt werden [3, S. 2].

Neben einem hybriden Ansatz für den Test besteht der Bedarf nach einer vertraglichen Festlegung des Testumfanges beziehungsweise eines potenziellen NDA für die Regelung der Handhabung sensibler Daten, ähnlich den weiteren Grenzen bei einem *Hardening Check*. Zusätzlich dient der Vertrag oder ein separates Dokument als eine Bestätigung der Rechtmäßigkeit des Penetrationstests, sollte von der Seite des *Providers*, *Microsoft*, oder bei einem Internetanbieter ein Problem vorliegen [3, S. 6].

Zuletzt besteht eine Vorgabe, jegliche erkannten Schwachstellen, die *Azure* und das *Azure Fabric* selbst betreffen, an *Microsoft* zu melden, ohne diese 90 Tage lang gegenüber einer anderen Partei preiszugeben. Hierfür besteht ein so genanntes *Bug Bounty*-Programm, bei dem etwaige Sicherheitslücken gemeldet und deren Auffinden finanziell gewürdigt wird [3, S. 6].

2.4 Microsoft Azure

Microsoft Azure, kurz *Azure*, ist *Microsofts* eigenes Cloud-Portfolio. Im Detail bedeutet dies, dass *Azure* eine Ansammlung von Serviceleistungen, Ressourcen und Funktionen darstellt, welche ortsungebunden angefordert und bedarfsgerecht skaliert werden können (siehe Kapitel 2, Abschnitt 2.1, Seite 6). Dieser Abschnitt gibt einen kurzen allgemeinen Überblick zu *Azure* und beleuchtet darüber hinaus verschiedene Architekturstile und Anwendungsbereiche.

2.4.1 Überblick

Azure definiert sich als „*Portfolio mit Clouddiensten*“. Diese Dienste unterliegen einer stetigen Weiterentwicklung und Ergänzung durch zeitgemäße Angebote, gemessen an dem technologischen Fortschritt in der IT-Welt. *Azure* dient dem Ziel, Organisationen bei der Bewältigung ihrer aktuellen und zukünftigen geschäftlichen Herausforderungen zu unterstützen. Insgesamt bietet Azure über 100 verschiedene Dienstleistungen für unterschiedliche Zwecke. Dieses Spektrum reicht von der simplen Installation und Ausführung von eigenen Anwendungen auf virtuellen Maschinen (VM) bis hin zu komplexen Gebilden wie beispielsweise maschinelles Lernen oder *Mixed Reality* [41].

Basis-Infrastruktur

Die Funktionsweise und Infrastruktur von *Azure* basiert auf vollständig auf Virtualisierung. Virtualisierung entkoppelt ein Betriebssystem von den physischen Ressourcen eines Computers mittels einer Zwischenebene, genannt *Hypervisor*. Im Fall von Azure werden Server mit je einem *Hypervisor* für das Betreiben „unzähliger“[4] VMs und anderer virtueller Services bereitgestellt [41].

Definition 2.11: Der *Hypervisor* (auch *Virtual Machine Monitor* (VMM)) ist eine abstrahierende Management-Software zwischen einem Host- und Gast-System. Er steuert die Zuweisung von Ressourcen zu und zwischen VMs und sorgt somit für einen reibungslosen Arbeitsablauf. Darüber hinaus verhindert der *Hypervisor* Behinderungen zwischen den einzelnen Gast-Systemen und reguliert ebenfalls die Wechselwirkungen zwischen Host und VMs, um diesen zu suggerieren, sie befänden sich auf einem vollwertigen physischen System. Zudem garantiert die Existenz eines *Hypervisors* ein gewisses Maß an Sicherheit, da die einzelnen Systeme nicht auf die Dateien des Host-Systems und untereinander auf die einer anderen VM zugreifen können [29].

[4] Dieser Wert ist abhängig von den Limitationen der Hardware

Diese Server werden in *Server Racks* kumuliert, welche wiederum in weltweit verteilten Rechenzentren gebündelt werden. Auf einem ausgewählten Server pro *Rack* wird eine spezialisierte Software, der *Fabric Controller* betrieben. Jegliche *Racks* eines Rechenzentrums sind über Switches miteinander vernetzt und die Rechenzentren stehen über das Internet in Verbindung. Je Rechenzentrum wird eine weitere spezialisierte Software ausgeführt, genannt *Orchestrator*, mit der jegliche *Fabric Controller* des Rechenzentrums kommunizieren. Dem *Orchestrator* fällt die Aufgabe zu, sämtliche Prozesse und Operationen *Azure* betreffend zu koordinieren. In diese Kategorie fallen beispielsweise Nutzeranforderungen über die Web-API (*Application Programming Interface*) des *Orchestrators*, mittels des so genannten *Azure Portals* [41]. Die bisher beschriebene Infrastruktur ist in Abbildung 2.6 exemplarisch dargestellt. Darüber hinaus zeigt Abbildung **??** eine beispielhafte Skizze der globalen Infrastruktur der *Azure Cloud*.

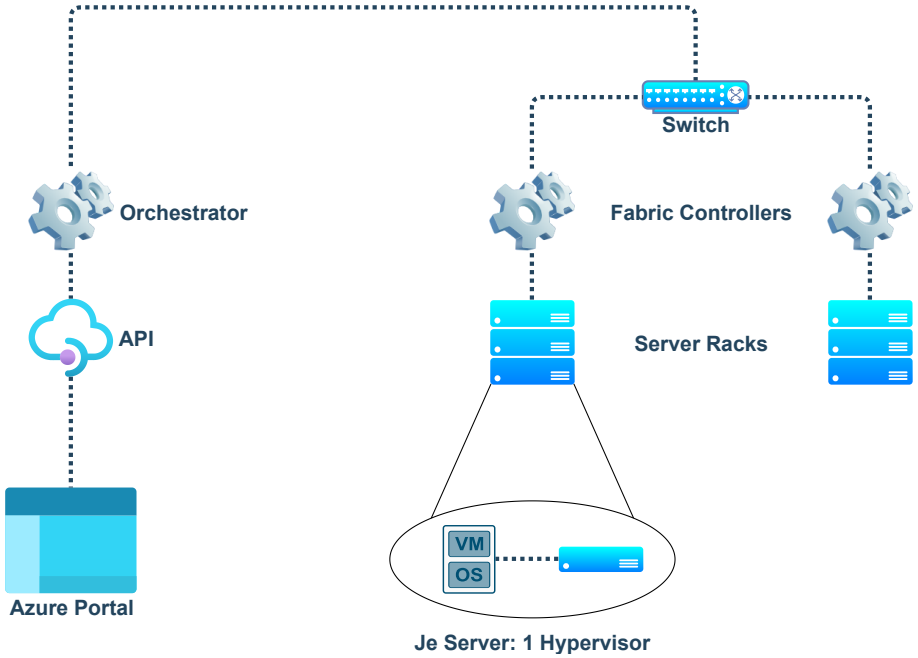

Abbildung 2.6: Exemplarische Infrastruktur eines Azure-Rechenzentrums [eigene Abbildung nach [41]]

Azure Portal

Das *Azure Portal* ist das webbasierte Management-Interface von *Azure*, welches mit einem registrierten *Azure*-Account verwendet werden kann, um Cloud-Services zu erstellen und zu verwalten (siehe Abbildung 2.7, Seite 33). Fordert ein Nutzer gemäß dem unter Basis-Infrastruktur im letzten Absatz erwähnten Beispiel hierüber eine VM an, gelangt diese Anforderung zum *Orchestrator* des im Konfigurationsprozess ausgewählten Rechenzentrums. Dieser erstellt ein Paket mit allen erforderlichen Informationen, wählt den bestgeeigneten *Server Rack* des Rechenzentrums (basierend auf dem Bedarf der gestellten Anforderung) und übergibt das generierte Informationspaket dem zugeordneten *Fabric Controller*. Dieser erstellt letztlich die VM, welche direkt im Anschluss von ihrem Nutzer verwendet werden kann [41].

Das *Azure Portal* stellt die Alternative zu den *Azure Command Line Tools* dar. Diese ermöglichen die Verwaltung aller *Azure*-Services von der Kommandozeile aus, während das Portal die webbasierte GUI-Variante bietet (*Graphical User Interface*). Im Detail ermöglichen diese beiden Services das Erstellen, Überwachen und Verwalten aller in *Azure* erzeugten Komponenten. Darüber hinaus gelingt im *Azure Portal* eine exakte Überwachung jeglicher Kosten und Ausgaben [41].

Gehostet wird das *Azure Portal* verteilt auf sämtliche *Microsoft*-Rechenzentren. Dies ermöglicht eine Redundanz gegenüber einzelnen Ausfällen, sodass das Portal stets erreichbar und verfügbar ist. Zudem werden so genannte *Bottlenecks* vermieden und jeder Kunde hat ungehinderten Zugriff auf den vollen Umfang der Funktionalitäten, ohne beispielsweise durch eine langsamere Internetanbindung gestört zu werden [41].

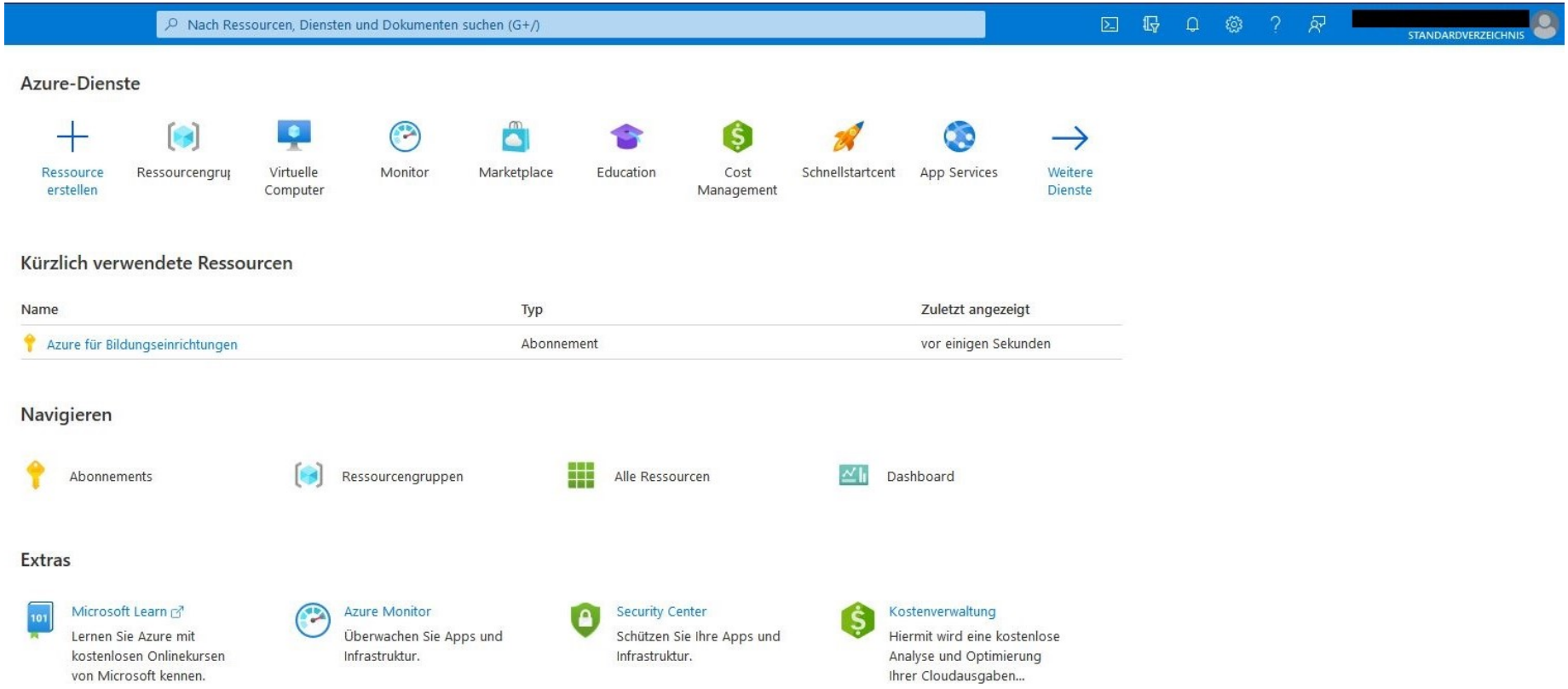

Abbildung 2.7: Azure Portal, anonymisiert [Screenshot]

2.4.2 Azure-Dienste

Gemäß den offiziellen Angaben von Microsoft besteht die *Azure Cloud*-Plattform aus mehr als 100 Produkten und Diensten [41]. Diese dienen alle einem anderen Zweck und können nach Belieben und den eigenen Anforderungen entsprechend erstellt und verwendet werden. Unterteilt werden die Services in zehn große Hauptkategorien, anhand derer ein genauerer Überblick über die *Azure*-Servicelandschaft ermöglicht wird. Diese Kategorien gliedern sich wie folgt (nach [40]):

Compute: Dies ist einer der häufigsten Einstiegspunkte in die Cloud, beispielsweise für Unternehmen. In diese Kategorie fallen sowohl Services für das Hosting von Anwendungen als auch von Rechnern selbst. Allgemein bedeuten alle Services die Bereitstellung und das Management von Rechenleistung für etwaige Aufgaben.

Netzwerk: Dienstleistungen dieser Kategorie dienen der Verbindung von Ressourcen und der Bereitstellung des Zugriffs auf diese. Somit sind *Azure*-Netzwerkdienste die Schnittstelle der Rechenzentren mit der Außenwelt.

Speicher: Einfaches Speichern von variablen Datenmengen auf verschiedene Art und Weise.

Mobile: Ermöglicht die Entwicklung und Bereitstellung von *Backend Services* für Mobil-Apps. Hierdurch soll die Einbindung von sensiblen Zusatzdiensten wie beispielsweise lokalen Ressourcen (SQL Server, SharePoint etc.; *Structured Query Language*) erleichtert werden.

Datenbanken: Hiermit gelingt die Bereitstellung beliebiger Datenbankdienste (SQL und NoSQL) für die Speicherung und Verwaltung von Daten jeglicher Datentypen. Diese Dienste zeichnen sich dank globalen Anschlussmöglichkeiten durch eine hohe Verfügbarkeit aus.

Web: Unterstützung bei dem Hosting und Erstellen von Webanwendungen und HTTP-basierten Webdiensten (*Hypertext Transfer Protocol*).

Internet der Dinge (IoT): Dienste, welche die Ansteuerung, Vernetzung und Kontrolle von internetfähigen Geräten ermöglichen und vereinfachen.

Big Data: Mit dieser Dienste-Sammlung unterstützt *Azure* die Verarbeitung und das allgemeine Arbeiten mit überdimensional großen Datenmengen, beispielsweise den Aufzeichnungen von Wetterstationen.

KI (Künstliche Intelligenz): Services dieser Kategorie ermöglichen den Einsatz von maschinellem Lernen in *Azure*. Hierbei sind die bekannten Anwendungsbereiche (Bildanalyse, Spracheingabe etc.) als so genannte *Cognitive Services* zur Verwendung bereitgestellt.

DevOps: Mithilfe der *DevOps*-Services gelingt die Automatisierung der Softwarebereitstellung über die geläufigen *CI/CD Pipelines* (*Continuos Integration, Continuos Delivery/Deployment*). Darüber hinaus wird die Bereitstellung von Infrastrukturen und Einbindung von Drittanbieterwerkzeugen, beispielsweise *Jenkins*, erleichtert.

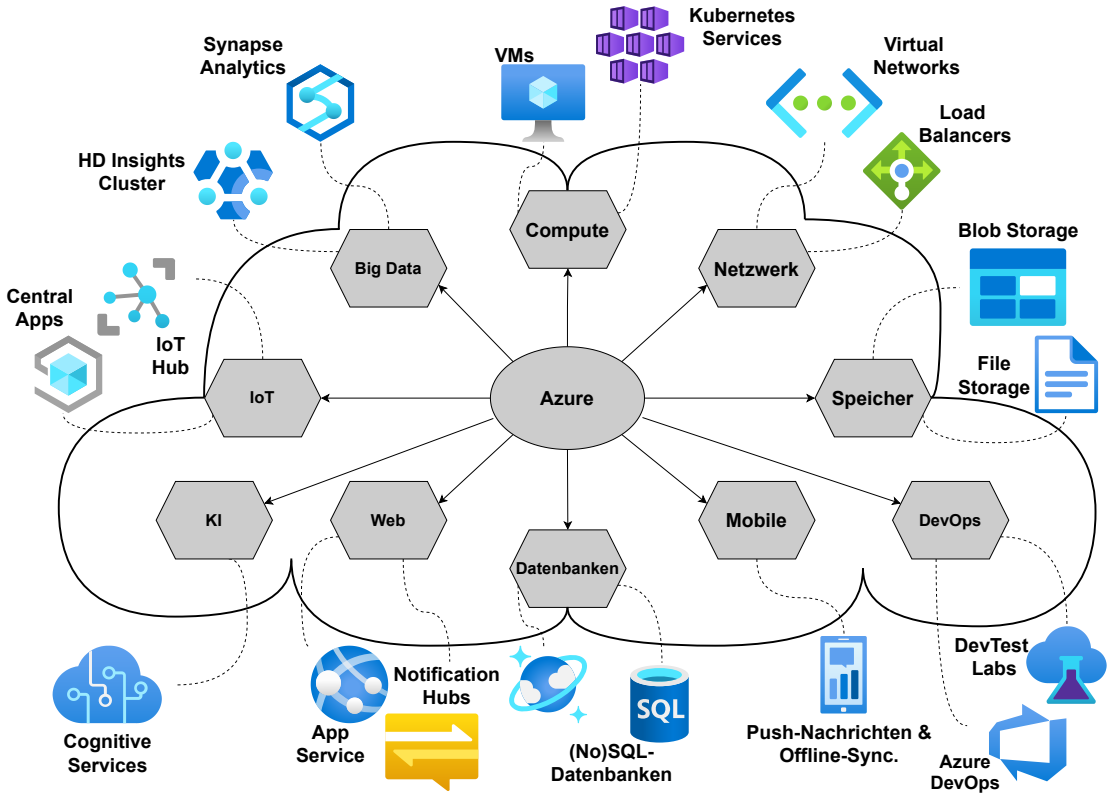

Abbildung 2.8: Kategorien der Dienstleistungen in Azure und Beispiele aus diesen [eigene Abbildung nach [40]]

2.4.3 Azure-Konten

Jeglicher Zugriff auf *Azure* und dessen Services beziehungsweise auf die Verwaltungsebene, das *Azure Portal*, erfordert ein so genanntes *Azure*-Konto. Innerhalb dieses Kontos wird mittels eines oder mehreren Abonnements die Erstellung und Verwendung von Cloud-Ressourcen verwaltet. Ein Abonnement fasst alle darunter gegliederten Anwendungen und Dienste innerhalb derselben Abrechnung zusammen. Beispielsweise wird einem Unternehmen so das parallele Anlegen mehrerer Abonnements für verschiedene Geschäftsbereiche ermöglicht, um diesen lediglich Zugriff auf die Anwendungen und Ressourcen zu erteilen, die für die jeweilige Arbeit erforderlich sind [28].

Zusätzlich lassen sich Abonnements in kleinere Einheiten, so genannte Ressourcengruppen einteilen. Diese Gruppen fassen einzelne Ressourcen, welche einem bestimmten Zweck dienen, zu einer gemeinsam kontrollierbaren Einheit zusammen. Dieser Sachverhalt ist in Abbildung 2.9 dargestellt. Ein Azure-Konto hat vollen Zugriff auf alle erstellten Abonnements und Ressourcengruppen [28]. Des Weiteren können *Azure*-Abonnements bei Bedarf und konformen Regeln und Richtlinien in so genannten Verwaltungsgruppen gebündelt werden. Dies ist dann nützlich und unter Umständen erforderlich, wenn die Anzahl der Abonnements eine sehr hohe Zahl erreicht und eine individuelle Verwaltung umständlich und wenig sinnvoll wird [18].

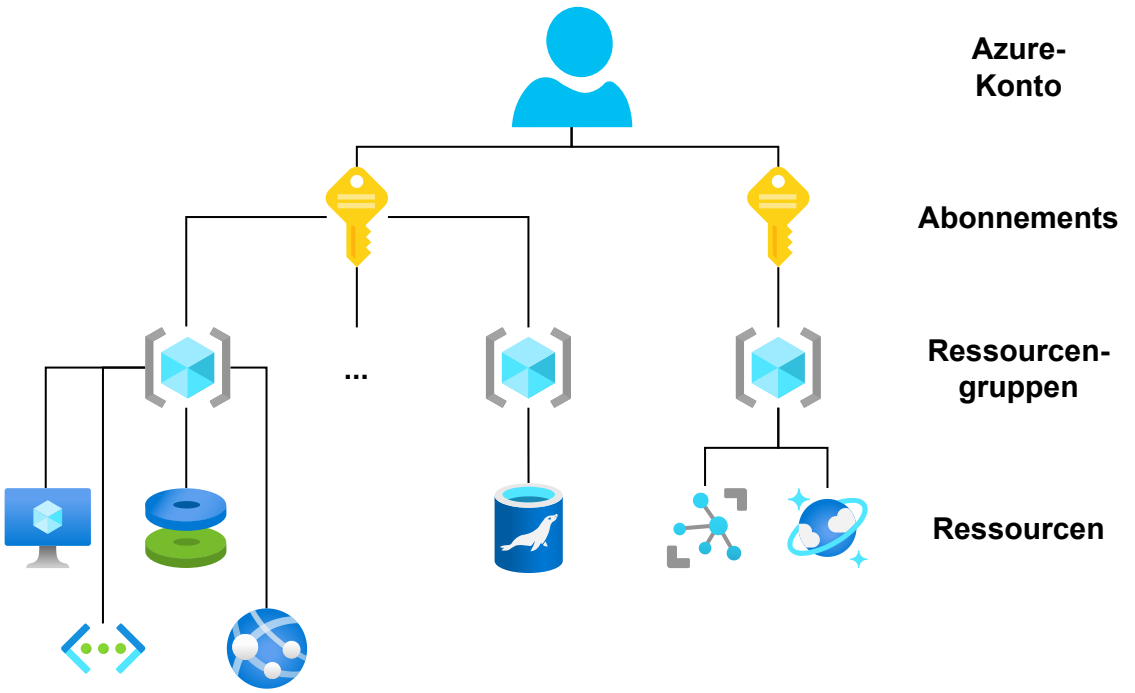

Abbildung 2.9: Exemplarische Darstellung der Aufteilung eines Azure-Kontos [eigene Abbildung nach [28]]

Im Zuge dieser Arbeit wurde im Rahmen des *Microsoft Azure for Students*-Programms ein kostenloser *Azure*-Account erstellt, um einen besseren Einblick in die Verwaltung und Nutzung etwaiger Dienste zu erhalten.

2.4.4 Azure Active Directory

Azure Active Directory (*Azure AD*) ist die Cloud-basierte Nutzerverwaltung und Autorisierungsschicht für jegliche *Azure Cloud*-Dienste. Hierbei ist klarzustellen, dass dieses AD keinen Ersatz für das *Windows Server Active Directory* sowie keine direkte Entsprechung darstellt. Im Grunde dient *Azure AD* dazu, an ein bereits vorhandenes lokales AD angeknüpft zu werden, um darin enthaltene Nutzer schnell in die *Azure Cloud* aufzunehmen, sodass diese mit denselben Anmeldeinformationen direkt auf dortige Ressourcen Zugriff haben und berechtigt sind, diverse Operationen durchzuführen [42].

Auch eine einzelne Verwendung von *Azure AD* ist umsetzbar. In kleineren Unternehmen, in denen kein lokales AD besteht, ist die *Azure*-Variante durchaus geeignet, als alleiniger Verzeichnisdienst den Zugriff auf gebuchte Cloud-Ressourcen zu ermöglichen. In diesem Szenario sollte beachtet werden, dass somit keine zentrale Nutzerverwaltung besteht. Die Authentifizierung auf lokalen Geräten mit lokalen Konten erfolgt standardmäßig mit Windows-Anmeldeinformationen [42].

Bei der Erstellung eines *Azure*-Kontos wird automatisch ein Standardverzeichnis mit erstellt. Dieses dient als Hauptinstanz des *Azure AD* und beherbergt sämtliche Benutzer, welche uneingeschränkten Zugriff auf alle Dienste der buchenden Institution erhalten. In der Fachsprache wird dieses Standardverzeichnis auch *Mandant* genannt. Dieser Begriff fast die zugehörige Institution (das zugehörige *Azure*-Konto) und deren Standardverzeichnis zusammen. Jedem Mandanten sind abgeschlossene *Azure*-Abonnements zugeordnet (siehe Abschnitt 2.4.3, Seite 35). Dies bedeutet, dass jedes Abonnement von einem einzelnen Konto besessen wird und somit folglich nur einem *Azure AD* zugeordnet werden kann. Dies bedeutet des Weiteren, dass demselben AD mehrere Abonnements zugeordnet werden können, jedoch nicht multiple AD zu einem Abonnement [42].

Im Falle einer Nutzerzuordnung zu einem oder mehreren Abonnements muss der Nutzer zwingend dem verwaltenden *Azure AD* bekannt sein. Sind beispielsweise zwei AD im Einsatz und ein Nutzer soll zu je einem Abonnement innerhalb dieser zugewiesen werden, muss der Nutzer beiden ADs bekannt sein. Mehrere *Azure AD*-Entitäten sind deshalb möglich, da ein Mandant zwar ein Standardverzeichnis aufweist, für Test- und Entwicklungszwecke jedoch weitere AD-Instanzen kreiert werden können. Hierfür ist jedoch die Besitzerrolle für ein *Azure*-Konto erforderlich [42].

Azure-Benutzerkonten im Unterschied zu Azure-Konten

Innerhalb eines Mandanten, bestehend aus *Azure*-Konto und AD-Instanz, müssen so genannte Benutzerkonten angelegt werden, um etwaige gebuchte Dienste zu nutzen. Diese Konten sind nicht mit einem *Azure*-Konto gleichzusetzen. Dieses ist der Schirmherr einer Organisation und wird zwar auch für den Zugriff auf *Azure* genutzt, der Hauptzweck liegt jedoch auf Seiten der Verwaltung. Benutzerkonten erlauben auf der anderen Seite kontrollierten und mittels Rechten und Rollen abgesicherten Zugriff auf eingerichtete *Azure*-Dienste. Hierbei enthält ein solches Konto die erforderlichen Anmeldeinformationen eines Nutzers sowie ein Regelset, mithilfe dessen der Zugriff und die operative Nutzung von Diensten kontrolliert wird [27].

Die Gesamtheit dieser Benutzerkonten lässt sich in drei Kategorien gliedern. Diese dienen der Unterscheidung in nur innerhalb des Mandanten existierende Benutzer, importierte beziehungsweise synchronisierte Identitäten und Gäste [27]. In der Tabelle 2.3 wird dieser Sachverhalt genauer dargestellt.

Tabelle 2.3: Kategorisierung von Benutzerkonten in Azure [eigene Tabelle nach [27]]

Identität	**Beschreibung**
Cloudidentität	Diese Art Benutzer existiert ausschließlich in *Azure AD*. In diesen Bereich fallen selbst erstellte Benutzer sowie Administrator-Konten. Das angelegte *Azure*-Konto beispielsweise ist automatisch ein „Globaler Administrator“ in der AD-Instanz und hat somit vollen Zugriff. Zudem können diese Benutzerkonten auch aus einem separaten *Azure AD* stammen. Bei einer dortigen Entfernung werden diese Konten gelöscht.
Mit AD synchronisierte Identität	Diese Identitäten entstammen einem lokalen AD (*Windows Server AD*) und werden bei der Synchronisierung eines solchen mit dem *Azure AD* übernommen.
Gastbenutzer	Externe Konten, beispielsweise von anderen Cloud-Anbietern oder externen Dienstleistern, fallen in diese Kategorie der *Azure*-Identitäten. Auch andere *Microsoft*-Konten können auf diese Weise temporär hinzugefügt werden.

3 Methoden und Vorgehensweise

Die definierten und erläuterten Grundlagen ebnen den Weg für die Methodiken und Vorgehensweise bei der Erstellung eines Konzepts für zwei Arten von Sicherheitstests in *Azure Cloud*-Infrastrukturen, Hardening und Penetrationstest. In diesem Kapitel wird die Vorgehensweise beschrieben, mithilfe derer die Ergebnisse dieser Arbeit entstehen sollen.

3.1 Informationsbeschaffung

Vor der Untersuchung und dem Vergleich verschiedener Vorgehensweisen für Sicherheitstests mussten Informationen beschafft werden. Hierbei wurde in mehreren Schritten vorgegangen, mit dem Ziel, für Penetrationstests und *Hardening Checks* mehrere etablierte Vorgehensweisen festzustellen und bezogen auf *Azure* zentrale Sicherheitsrisiken und zu berücksichtigende Faktoren herzuleiten.

Begonnen wurde mit der Beschaffung und Durchsicht verschiedener themenbezogener Quellen. Zu Beginn lag das Ziel vor allem darin, wichtige Grundlagen für die Themen *Cloud Computing* und *Azure* zu erarbeiten und zu definieren. Darüber hinaus spielte die Erläuterung von Sicherheitstests in der IT sowie die Einordnung von Penetrationstests und *Hardening Checks* in diesen Rahmen eine zentrale Rolle. Diesen drei dargestellten Aspekten widmet sich das Kapitel Grundlagen.

Primär wurde für den Bezug der einzelnen Quellen die Suchmaschine *Google* beziehungsweise deren wissenschaftsorientierter Ableger *Google Scholar* genutzt. Weiterhin wurde die bibliothekseigene Online-Suche der Hochschule Mittweida, *Primo*, genutzt, um neben digitalen Papern auch zu berücksichtigende Buchquellen herauszufinden. Allgemein erstreckte sich die durchgeführte Recherche über das Feld *Microsoft Azure*. Spezifisch wurde mit der Suche nach den Begriffen **Cloud Computing** und **Cloud-Infrastruktur** begonnen. Dies brachte verschiedene Online-Artikel und Ergebnisse aus Wissensspeichern, welche den Grundstein für die Erarbeitung des ersten Wissens zu Eigenschaften, Service-Modellen und Deployment-Modellen in der Cloud legten. Anschließend wurden darauf aufbauend die Begriffe **Sicherheitstests**, **Penetrationstest** und **Hardening** recherchiert, um Informationen über die Definitionen dieser Tests und den zugrundeliegenden Aufbau zu gewinnen. Basierend darauf fand eine Auseinandersetzung mit dem allgemeinen Aufbau, wichtigen Teilaufgaben und zentraler Aspekte dieser Arten von Sicherheitstests statt, um eine Betrachtung von Sicherheitstests in der IT sowie eine Einordnung der gewählten Unterarten vorzunehmen.

Anschließend wurde zur *Azure Cloud* recherchiert, um Hintergrundinformationen zu deren zentraler Funktionsweise und Aufbau zu erhalten. Hierbei lag der Fokus insbesondere auf den Suchbegriffen **Azure Server**, **Azure Rechenzentrum**, **Azure Funktionsweise** und **Azure Services**. Aus den ermittelten Informationen wurde eine Übersicht der Struktur eines *Azure*-Rechenzentrums sowie eine exemplarische Struktur der globalen *Azure*-Infrastruktur erstellt. Zudem konnten somit die zentralen Funktionen des *Azure Portal*, *Azure Marketplace* und die zahlreichen Services und Funktionen von *Azure* erschlossen und beschrieben werden. Basierend hierauf wurde außerdem anhand der Suchbegriffe **Azure Angriffsfläche**, **Azure Sicherheitsrisiken** und **Azure Sicherheit Schwachstellen** versucht, die potenzielle Angriffsfläche zu ermitteln. Im Detail bedeutete dies die Überprüfung von Angriffsvektoren und damit assoziierten Angreifern und exemplarischen Angriffen, welche auf diesen Wegen durchgeführt werden könnten.

Nachdem wichtige allgemeine und spezifische Grundlagen zu den Themen *Cloud Computing*, Cloud-Infrastruktur, Sicherheitstests und *Azure Cloud* erarbeitet wurden, wurde die Recherche mit dem Fokus auf Vorgehensweisen für Penetrationstests und *Hardening Checks* speziell in *Azure* fortgesetzt. Hierbei fand eine Suche nach den Begriffen **Azure Hardening**, **Azure Penenetration Testing**/**Pentesting**, **Azure Security** sowie **Azure Hardening-Standards** statt. Infolgedessen wurden in der Kategorie *Hardening* drei Vorgehensweisen beziehungsweise Standards ausgewählt, welche beschrieben und deren Inhalt im Detail dargestellt werden sollten. Auf dieselbe Weise wurde für Penetrationstests verfahren, jedoch wurden hier lediglich zwei Vorgehensweisen identifiziert und ausgewählt. Die Auswahl dieser fünf Standards und Vorgehensweisen für beide Arten von Sicherheitstests stellt das Fundament für die Beschreibung, den Vergleich und die Bewertung dieser dar. Zusätzlich hierzu wurde nach den Begriffen **Gesetze IT-Sicherheit**, **Rahmenbedingungen Hardening**/**Pentesting** gesucht sowie bisherige Quellen betrachtet, um etwaige Rahmenbedingungen, Gesetze oder technische und fachliche Anforderungen zu erkennen, welche für die Durchführung eines Sicherheitstests in *Azure* eine Rolle spielen. Zuletzt wurde diese Suche auf empfehlenswerte **Werkzeuge**/**Tools** ausgeweitet, die in einem der beiden Fälle oder in beiden genutzt werden können oder empfehlenswert sind.

Die recherchierten Informationen und Quellen lieferten die erforderlichen Informationen, um die Ergebnisse dieser Arbeit, welche Inhalt des Kapitels Ergebnisse sind, zu innovieren.

3.2 Analyse der Angriffsfläche

Die Struktur der Ergebnisse sollte nach folgendem Muster aufgebaut werden: Feststellung der möglichen Angriffsfläche, Präsentation lohnenswerter Vorgehensweisen für Sicherheitstests zur Problemerkennung, Empfehlung von Vorgehensweisen für verschiedene Szenarien sowie Empfehlung der Sicherheitstests für einzelne Service-Modelle.

Begonnen wurde mit einer Zusammenstellung von bekannten und potenziellen Angriffsvektoren auf die *Azure*-Infrastruktur. Vielmehr sollten hierbei auch Arten von Angreifern, welche mit diesen Vektoren korrespondieren, bekannte potenziell durchführbare Angriffe und ausnutzbare Sicherheitsrisiken herausgearbeitet werden. Dies wurde wie beschrieben umgesetzt, wodurch eine Übersicht der Angriffsfläche bei *Azure* entstanden ist. Die einzelnen Vektoren haben verschiedene Teile der *Azure*-Hard- und -Software zum Ziel, welche gleichzeitig unterschiedliche Arten von Angreifern für jeden Vektor bedingen. Somit ergeben sich verschiedene einzigartige Bedrohungsszenarien, welche durch differenzierte Angriffsszenarien manifestiert werden. Diese Angriffsfläche (siehe Kapitel 4, Abschnitt 4.1, Seite 45) befindet sich zudem in einem ständigen Wandel, ausgelöst durch unterschiedliche zu berücksichtigende Faktoren. Aus diesem Grund wurden sowohl spezielle Risiken, welche durch die technische Realisierung von *Azure* entstehen, beispielsweise in Zusammenhang mit dem *Hypervisor* sowie traditionelle Bedrohungen der IT-Sicherheit berücksichtigt. Die einzelnen Angriffsvektoren und die Komponenten, auf welche diese innerhalb eines Cloud-Knotens zielen, wurden grafisch verdeutlicht. Anschließend wurde jeder Vektor kurz erläutert sowie der zugehörige Angreifertyp dargestellt. Diese Erklärungen dienen der Darstellung potenzieller Auswirkungen eines Angriffs auf diesem Weg sowie des Schadens, welche somit angerichtet werden könnte.

3.3 Beschreibung und Bewertung der Vorgehensweisen

Anschließend wurden die Vorgehensweisen für *Hardening Checks* und Penetrationstests getrennt voneinander beschrieben. Zuvor fand eine Darstellung des jeweiligen Testverfahrens sowie eine Begründung für dessen Nutzung statt. Für die Erläuterung der Vorgehensweisen stand die Verdeutlichung der einzelnen konkreten Inhaltspunkte und deren Bedeutung beziehungsweise Zielstellungen im Vordergrund, um einen Überblick zu liefern, welche Ergebnisse bei der Umsetzung oder Durchführung der jeweiligen Vorgehensweise erzielt werden können. Hierzu wurden die einzelnen Werke textuell erläutert und anschließend über ein Ablaufdiagramm mit einer stichpunktartigen Zusammenfassung ihres Inhalts visualisiert. Weiterhin ist eine Empfehlung anhand bestimmter Kriterien vorgenommen worden, um einen Anhaltspunkt zu liefern, wann welche Vorgehensweise für einen *Hardening Check* oder Penetrationstest sinnvoll eingesetzt werden kann. Anhand dieser ist es möglich, zu identifizieren, um welche Art Vorgehensweise es sich handelt, um situationsgemäß über eine Verwendung entscheiden zu können. Zwischen *Hardening Check* und Penetrationstest unterscheiden sich die gewählten Kriterien geringfügig. Folgende Tabelle 3.1 stellt die jeweiligen Aspekte gegenüber und die spezifischen Unterschiede dar.

Tabelle 3.1: Gegenüberstellung der Bewertungsaspekte für die Vorgehensweisen bei Hardening Checks und Penetrationstests [eigene Tabelle]

Kriterien Hardening		**Kriterien Penetrationstest**
Technische Tiefe		Technische Tiefe
Organisatorische Tiefe		Organisatorische Tiefe
Nutzung nativer Technologien	**vs.**	Serviceorientierung
Industrieller Standard		Schwachstellenorientierung
Sicherheitsniveau		Provider
Automatisierbarkeit		Testumfang

Sowohl die Vorgehensweisen für *Hardening Checks* als auch Penetrationstests werden jeweils untereinander hinsichtlich ihrer technischen und organisatorischen Tiefe verglichen. Nachfolgend weichen die gewählten Kriterien jedoch voneinander ab, da die *Hardening*-Anweisungen in ihrer Nutzung von nativen Sicherheitstechnologien sowie deren Status als industrieller Standard und die Penetrationstest-Guidelines bezüglich ihrer Orientierung nach Cloud-Services oder potenziellen Schwachstellen verglichen werden. Zuletzt wird die Automatisierbarkeit beziehungsweise respektive der Testumfang überprüft.

Technische Tiefe bezeichnet den Detailgrad technischer Beschreibungen. Dies bezieht sich auf genaue Durchführungsanweisungen für etwaige Testfälle sowie auf mögliche Begründungen oder Beschreibungen, warum ein Testfall durchgeführt werden sollte oder was eine bestimmte Konfigurationseinstellung bedeutet.

Organisatorische Tiefe bezeichnet den Detailgrad organisatorischer Beschreibungen. Hierbei handelt es sich um Anweisungen, wie ein Test (*Hardening Check*, Penetrationstest) aufzubauen ist oder welche Parameter bei der Planung, Durchführung oder Nachbereitung zu berücksichtigen sind.

Nutzung nativer Technologien bezeichnet das Vorhandensein von Anweisungen oder Erklärungen zu nativen Sicherheitstechnologien in *Azure*, welche während eines *Hardening Checks* überprüft werden oder die Grundlage für die Härtung des *Azure*-Mandanten bilden. Dies bezieht sich auf Sicherheitslösungen, welche *Microsoft* selbst für die Garantie oder Optimierung der Sicherheit in *Azure* bereitstellt.

Industrieller Standard trifft eine Aussage darüber, ob die betrachtete Vorgehensweise ein etablierter, anerkannter Industriestandard ist, nach welchem beispielsweise eine Zertifizierungsstelle Überprüfungen vornehmen würde.

Sicherheitsniveau gibt an, welche Sicherheitsstufe das Treffen einer Konfigurationseinstellung oder erfolgreiche Überprüfen und Einrichten einer Kontrolle aus einem *Hardening*-Standard zur Folge hat. Hier wird differenziert zwischen einem Grund-Sicherheitsniveau und einem Hochsicherheitsniveau.

Automatisierbarkeit bezeichnet Möglichkeiten, die existieren, die Umsetzung und Überprüfung eines *Hardening*-Standards und dessen Kontrollen teilweise automatisiert vorzunehmen, beispielsweise über Skripte oder andere Software.

Serviceorientierung gibt Auskunft darüber, ob die zu bewertenden Vorgehensweisen für Penetrationstests in *Azure* einen Fokus auf bestimmte Services legen, welche in jedem Fall getestet werden sollten.

Schwachstellenorientierung erfüllt dieselbe Funktion wie Serviceorientierung und bewertet die Vorgehensweisen in Frage auf ihren Fokus hinsichtlich unterschiedlicher potenzieller Schwachstellen beziehungsweise Schwachstellenkategorien.

Provider erfasst die Ausrichtung einer Vorgehensweise daraufhin, ob die behandelten Testfälle lediglich auf einen Provider (in diesem Fall *Azure*) oder universell anwendbar sind.

Testumfang bewertet die Inklusivität der Vorgehensweise für Penetrationstests hinsichtlich ihrer Abdeckung. Hierbei wird unterschieden, ob lediglich die wichtigsten Services oder universell möglichst viele Testfälle berücksichtigt werden.

Neben der Empfehlung innerhalb der jeweiligen Arten von Sicherheitstests wurde abschließend eine Betrachtung der Anwendbarkeit von *Hardening Checks* oder Penetrationstests bei *Azure* in Abhängigkeit des zu verifizierenden Service-Modells durchgeführt. Hierbei fokussiert sich dieser Teil der Ergebnisse auf die Bewertung des Nutzens und der Durchführbarkeit. Berücksichtigt werden sich differenzierende Gegebenheiten bezüglich der Verantwortung und Kontrolle über eine Cloud-Infrastruktur, welche sich zwischen dem *Cloud Provider* und Kunden variabel verschieben. Abschließend wurden die einzelnen Vorgehensweisen innerhalb ihrer Testarten untereinander verglichen und bewertet. Ebenso wurde die Aussagekraft der Anwendbarkeitsbewertung betrachtet.

4 Ergebnisse

In diesem Kapitel werden die erlangten Ergebnisse dieser Arbeit dargestellt. Hierbei untergliedern diese sich in eine Präsentation der potenziellen Angriffsfläche bei *Azure*, die ausgewählten bekannten Vorgehensweisen für einen *Hardening Check* und die ausgewählten bekannten Vorgehensweisen für einen Penetrationstest in *Azure*. Den Beschreibungen dieser folgen zwei Abschnitte mit Einsatzempfehlungen.

4.1 Erfassung potenzielle Angriffsfläche

Um die Sicherheitslage einer Cloud-Infrastruktur im Allgemeinen richtig einschätzen zu können, sind Erkenntnisse darüber, wie diese angegriffen werden kann, zentral. Dies gelingt über eine Feststellung der Angriffsfläche. Hierbei wurde der Versuch unternommen, herauszuarbeiten, welche Wege ein Angreifer einschlagen könnte sowie welche Arten von Bedrohungen für die Sicherheit einer Cloud-Infrastruktur generell bestehen. Aus diesen so genannten Angriffsvektoren lassen sich potenzielle Sicherheitsrisiken ableiten, welche in einem Cloud-Umfeld bestehen.

Begonnen wird mit einer Kategorisierung möglicher Angriffsvektoren. Laut dem Glossar des ISTQB (*International Software Testing Qualifications Board*) bezeichnet ein Angriffsvektor den Weg eines Angreifers, maliziösen Zugriff auf ein IT-System zu erlangen [16]. Zudem ist das Modell des *Cloud Computing* einer Reihe von Sicherheitsrisiken ausgesetzt, welche durch diese Vektoren verursacht werden [5]. Abschließend wird Bezug auf verschiedene Angriffsszenarien genommen, welche schlussendlich durch die Vektoren und Risiken ermöglicht werden.

4.1.1 Angriffsvektoren

Um potenziell schädliche Aktionen in einer Cloud-Infrastruktur durchzuführen, bieten sich verschiedenen Angreifern unterschiedliche Angriffsvektoren. Hierbei besteht grundlegend ein Zusammenhang zwischen drei Typen von Angreifern und respektive drei Arten von Angriffsvektoren. Die Vektoren werden wie folgt benannt: Netzwerk, *Hypervisor* und Rechenhardware (*Compute*). Bei den Angreifern handelt es sich auf der anderen Seite um externe oder interne Nutzer sowie den *Cloud Provider* selbst oder spezifischer ausgedrückt einen Mitarbeiter mit unlauteren Zielen [5]. Die folgende Abbildung 4.1 veranschaulicht die erwähnten Angriffsvektoren als rote Blitze sowie die darüber zu erreichenden Zielkomponenten als Richtungspfeile an einem exemplarischen Server eines Rechenzentrums.

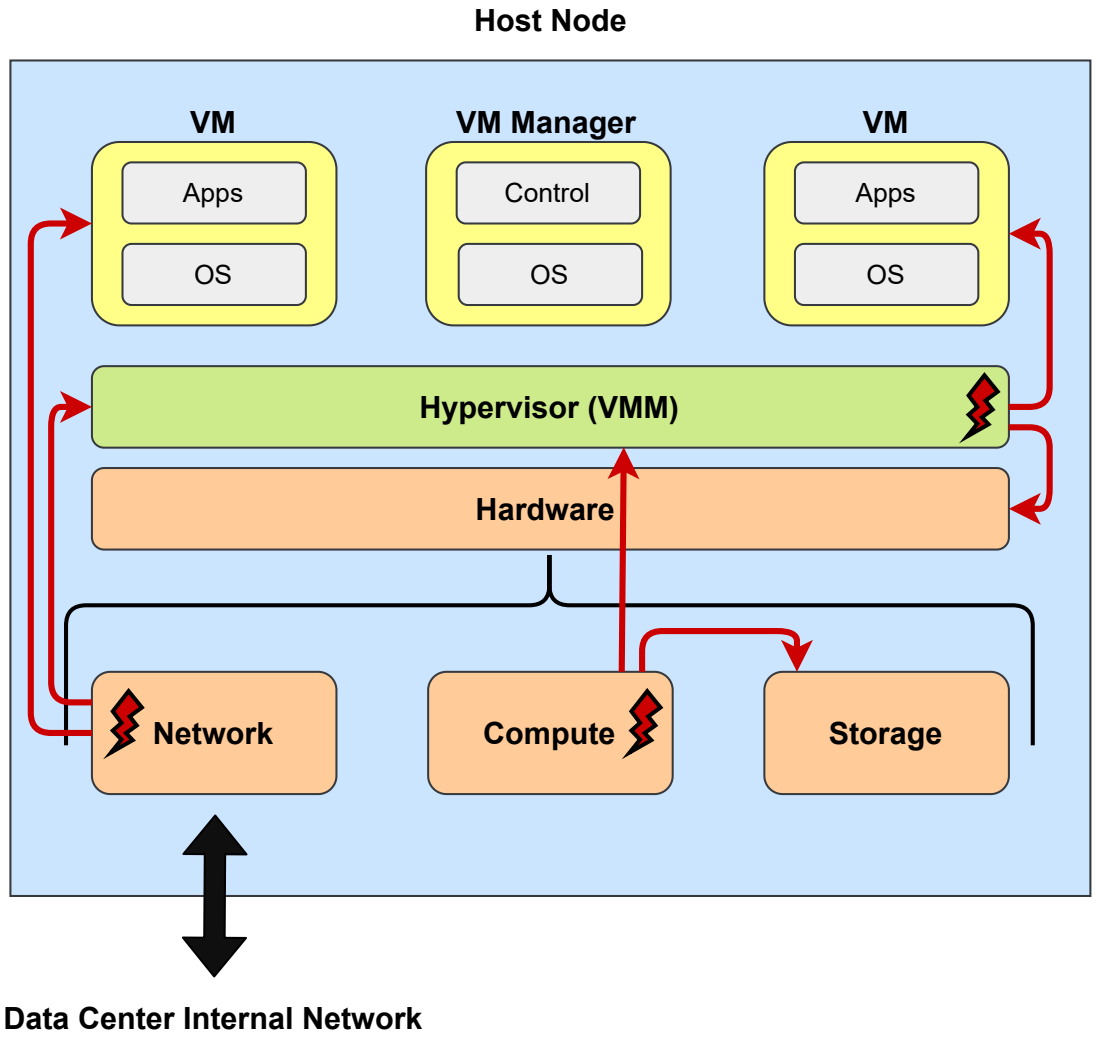

Abbildung 4.1: Visualisierung der Angriffsvektoren und ihrer einzelnen Ziele innerhalb eines Cloud-Knotens [eigene Abbildung nach [5]]

Netzwerk: externe Nutzer

Der Vektor Netzwerk wird von externen Nutzern für einen Angriff genutzt. Hierüber besteht die Möglichkeit, Vertraulichkeit und Integrität von Datenströmen einzelner Cloud-Anwendungen oder Infrastrukturen zu beeinträchtigen. Darüber hinaus ist auch eine negative Auswirkung eines Netzwerkangriffs für die Rechenzentren des *Cloud Providers* denkbar. Hier könnte beispielsweise die Verfügbarkeit eingeschränkt werden. Dieser Vektor kann den Grundstein für Folgeangriffe legen, wenn über das Netzwerk Zugriff auf ein Zielsystem erlangt werden kann [5].

Dieser Vektor ist die Schnittstelle von *Azure* mit der Außenwelt und der einzige Weg, auf einen Mandanten zuzugreifen. Zwar können hier verschiedene Protokolle genutzt werden, jedoch geschieht jegliche Kommunikation mit Services, Ressourcen oder dem *Azure*-Konto selbst auf diesem Weg. Aufgrund dessen kommt dem Netzwerk eine be-

sondere Bedeutung bei der Überprüfung und Aufrechterhaltung der Sicherheit in *Azure* zu. Dieser Umstand unterscheidet sich über die drei Service-Modelle SaaS, PaaS und IaaS nicht. Einzig der Umfang, in dem ein Kunde die Netzwerkschnittstellen beziehungsweise -komponenten verwalten kann, ändert sich (siehe Abbildung 2.1, Seite 11).

Hypervisor: interne Nutzer

Interne Nutzer führen Angriffe über den Hypervisor aus. Hierbei handelt es sich um Kunden des *Cloud Providers*, welche dessen Dienste in Anspruch nehmen, beispielsweise VMs. Ermöglicht wird die Ausnutzung des *Hypervisors* als Angriffsvektor über das Prinzip der *Multi-Tenancy*, auf dem *Cloud Computing* grundlegend aufbaut. Angreifer und Opfer teilen denselben physischen Host, sind jedoch voneinander isoliert. Diese Isolation leidet in Multi-Tenant-Architekturen jedoch unter vielseitigen Sicherheitsproblemen, insbesondere auf Hardwareebene. Hier ist die Trennung einzelner Ressourcennutzer nicht ebenso vollständig möglich, wie beispielsweise auf Applikationsebene. Demzufolge können Angriffe über den *Hypervisor* Vertraulichkeitsverluste sensibler Daten nach sich ziehen. Vielmehr ist es nicht möglich, diesen Vektor restlos zu schließen [5].

Die Kontrolle über den *Hypervisor* als Komponente unterliegt vollständig dem *Cloud Provider*, in diesem Fall *Microsoft*. Unabhängig vom Service-Modell kann dieser niemals in einen *Hardening Check* oder Penetrationstest mit einbezogen werden. Einzig im Fall eines *On Premise*-Deployment, bei welchem der Kunde das so genannte *Azure Fabric* selbst in einem Rechenzentrum hostet, wäre dieser Angriffsvektor zu berücksichtigen.

Compute: Cloud Provider

Der letzte Angriffsvektor konstituiert sich aus den Computing-Ressourcen (Rechenressourcen) der Cloud. Diese könnten potenziell von maliziösen Akteuren innerhalb der Reihen des Providers, beispielsweise Mitarbeiter, dazu genutzt werden, sensible Daten von Kunden zu stehlen oder physische und logische Hardware mit Schadenswirkung zu manipulieren. Ermöglicht wird dies durch das erforderliche Vertrauen, welches der Kunde in den *Cloud Provider* investiert [5].

Ähnlich dem *Hypervisor* besteht auch hier weder bei SaaS, PaaS noch bei IaaS eine Kontrolle vonseiten des Kunden über diese Ressourcen. Dies bedeutet, dass diese niemals in einen Sicherheitstests mit einbezogen werden können und die Betreuung dieses Angriffsvektors dem *Provider* zufällt. Die Ausnahme dieser Regel ist ebenfalls ein *On Premise*-Szenario.

Zentrale Bedeutung des Netzwerkvektors

Das Etablieren von Sicherheitsmaßnahmen in komplexen Umgebungen wie Cloud-Infrastrukturen bringt trotz des zentralen Managements von Ressourcen vielfältige Hürden und Risiken mit sich. Beispielsweise werden Daten an (unbekannten) externen Orten gespeichert und verwaltet. Die potenziell sensible Natur dieser Daten verdeutlicht den zugehörigen hohen Schutzbedarf [4, S. 7]. Das Netzwerk als Schnittstelle zwischen diesen Orten und dem Kunden ist aus diesem Grund als wichtigster Angriffsvektor zu betrachten und ein besonderes Augenmerk auf dessen Überprüfung zu lenken.

Zwar teilen sich nicht in weiterer Beziehung stehende Kunden des *Cloud Providers* dieselben Ressourcen und insbesondere bei der *Public Cloud* als Deployment-Modell zeichnen sich entsprechend hohe Risiken ab [4, S. 7]. Diese Problematik, ausgelöst durch den *Hypervisor* und die geteilten *Computing*-Ressourcen, kann jedoch durch den Kunden nicht kontrolliert werden, sondern entfällt bei einem Cloudeinsatz, welcher nicht *On Premise* stattfindet, *Plublic* oder *Private*, stets auf den *Provider*.

Ein Bericht der *Cloud Security Alliance* (CSA) aus dem Jahr 2020 beschreibt eine Bewegung der Sicherheitsrisiken im *Cloud Computing* weg von den Bedrohungen, welche in die Verantwortung des Providers fallen (z.B. (D)DoS), zu Risiken, welche auf der Seite des Kunden in Betracht gezogen und mitigiert werden müssen. Folgende Tabelle 4.1 stellt die hierbei referenzierten Top elf Risiken laut CSA dar. Hierzu wurden 241 IT-Sicherheitsexperten zu ihrer Meinung interviewt [2].

Die Mehrzahl dieser genannten Bedrohungen könnte potenziell über den Angriffsvektor Netzwerk herbeigeführt beziehungsweise ausgenutzt werden. Die Aspekte, welche von dieser Einschätzung betroffen sind, wurden in der Tabelle rot markiert.

Tabelle 4.1: Top Sicherheitsrisiken im Cloud Computing [eigene Tabelle nach [2]]

Nr. (alt)	Bedrohung	Übersetzung
1 (1)	Data Breaches	Datenverluste/Datenschutzverluste
2	Misconfiguration & Inadequate Change Control	Fehlkonfiguration & unzureichende Änderungsüberwachung
3	Lack of Cloud Security Architecture & Strategy	Fehlende Sicherheitsarchitektur/-strategie
4	Insufficient Identity, Access, Credential & Key Management	Unzureichendes Authentifizierungsmanagement (Identität, Zugriff, Credentials, Schlüssel)
5 (5)	Account Hijacking	Böswillige Accountübernahme
6 (6)	Insider Threat	Interne Bedrohung (z.B. Mitarbeiter)

Fortsetzung von Tabelle 4.1

7 (3)	Insecure Interfaces & APIs	Unzureichende gesicherte Interfaces & APIs
8	Weak Control Plane	Schwache Steuerungsebene
9	Metastructure & Applistructure Failures	Meta- & App-Strukturschwächen, Ausfälle
10	Limited Cloud Usage Visibility	Wenig Kontrolle und Management von Cloudnutzung
11 (11)	Abuse & Nefarious Use of Cloud Services	Missbrauch und Bösartiger Einsatz von Cloud-Services

Laut CSA spielen herkömmliche Bedrohungen zwar noch eine Rolle, waren in der Befragung jedoch so wenig vertreten, dass eine Abbildung in der Tabelle nicht lohnenswert war. Dies betrifft insbesondere Schwachstellen basierend auf geteilten Ressourcen, DoS-Angriffe sowie Datenverluste seitens des Providers. Neue Entwicklungen in diesem Bereich sind höher im Technologiestapel anzuordnen, spezifisch in Bereichen, welche von Kundenentscheidungen beeinflusst werden [2].

4.1.2 Beispielhafte Angriffsmethoden über das Netzwerk

Netzwerkbasierte Angriffe stellen den Hauptteil der Angriffe gegen Cloud-Infrastrukturen dar. In diese Kategorie fallen vor allem Angriffe, welche bereits von gewöhnlichen Infrastrukturen bekannt sind. Dies sind beispielsweise (D)DoS-Angriffe, Spoofing- und Sniffing-Angriffe auf Protokolle wie IP, ARP oder SOAP (*Internet Protocol, Address Resolution Protocol, Simple Object Access Protocol*) sowie *Code Injection*-Angriffe. Letztere beinhalten XSS (*Cross-Site Scripting*), *HTML Injection* (*Hypertext Markup Language*) oder *SQL Injection* [5]. Folgende Tabelle 4.2 verdeutlicht die Ziele der einzelnen Angriffsmöglichkeiten.

Tabelle 4.2: Netzwerkbasierte Angriffe und deren Ziele [eigene Tabelle nach [5]]

Angriff	**Ziel**
(D)DoS	Netzwerk-/Transportebene; Limitierung der Verfügbarkeit von Cloud-Services
IP/ARP/SOAP Spoofing/Sniffing	Angriff gegen Integrität und Vertraulichkeit von Daten; Mitlesen von Netzwerkverkehr und Erfassen bzw. Stehlen sensibler Daten (Nutzerdaten, Bankdaten etc.)

Fortsetzung von Tabelle 4.2

Code Injection (XSS, HTML, SQL)	Ausnutzen solcher Schwachstellen ermöglicht das Ausführen des betreffenden Codes mit den Privilegien der angegriffenen Anwendung; unerlaubter Zugang zu den Ressourcen und Anwendungen eines (anderen) Cloud-Nutzers

4.2 Azure Hardening

Sicherheit ist eine zentrale Anforderung an ein gehärtetes System. Dies bedeutet Widerstandsfähigkeit gegen negative Einflüsse von außen und innen, speziell fokussiert auf eine Minimierung von bestehenden Sicherheitsrisiken. Zuletzt wird einhergehend mit dem Faktor Sicherheit erwartet, dass ein gehärtetes System diesen Zustand zukunftssicher über die Zeit beibehalten kann, ohne Hard- oder Software von Grund auf ändern zu müssen [13, S. 61].

Allgemein ausgedrückt ist das Thema Sicherheit in *Azure* differenziert zu betrachten. Spezifischer formuliert hängt die Sicherheit immer von der Anwendung oder dem Service selbst ab, welche in *Azure* ausgeführt werden soll beziehungsweise wie die Infrastruktur einer Anwendung aufgebaut ist. Hierbei sollten abwägend Vor- und Nachteile von Ressourcen und Sicherheitseinrichtungen mit Blick auf den Nutzen für die zu schützende Architektur betrachtet werden. Potenzial für einen Kontrollverlust über die eigene Infrastruktur ist auch in der Cloud nach wie vor greifbar [45, S. 7].

4.2.1 Bekannte Vorgehensweisen

Für das *Hardening* von *Azure Cloud*-Infrastrukturen existieren verschiedene Vorgehensweisen, denen gefolgt werden kann beziehungsweise die als Standards im Feld der Informationssicherheit anerkannt sind. Diese zeigen kategorisch Kontrollpunkte auf, beispielsweise in Form von Konfigurationseinstellungen, welche automatisiert oder manuell verifiziert werden können, um ein maximales Sicherheitsniveau herzustellen. Für diese Arbeit wurden drei Vorgehensweisen ausgewählt, welche nachfolgend kurz vorgestellt und erläutert werden. Hierbei handelt es sich um das Buch **Mastering Azure Security** von Mustafa Toroman und Tom Janetscheck, den **Azure Security Benchmark** von *Microsoft* selbst sowie den **CIS Microsoft Azure Foundations Benchmark** des *Center for Internet Security* (CIS). Die Grunddefinition eines *Benchmarks* ist eine Vergleichsgrundlage für verschiedene Leistungen. In der IT-Sicherheit bezeichnet ein *Benchmark* eine Konfigurationsbeschreibung eines IT-Systems, welche die bestmögliche Sicherheit bietet [22].

Mastering Azure Security

Toroman und Janetscheck thematisieren die Sicherstellung von bestmöglicher Sicherheit in *Azure* mithilfe von der Plattform selbst bereitgestellten Hilfsmitteln. Das *Azure Security Center* und dessen Funktionen sowie nützliche Cloud-Services mit dem Fokus auf Sicherheit werden hierbei vorgestellt und deren Einsatz erklärt. Ziel des Buches ist die Schaffung eines Verständnisses für Gefahrenerkennung und deren zielführende Bekämpfung in *Azure* sowie die Vermittlung der Fähigkeit, sichere *Azure*-Infrastrukturen zu konzipieren [52, S. vii].

Die von Toroman und Janetscheck vorgestellte Methode zur Sicherung von *Azure* bezieht sich auf insgesamt acht zentrale Punkte, welche in Kombination ein ganzheitliches *Hardening*-Konzept ergeben, mit dessen Hilfe letzten Endes eine wirkungsvolle Sicherheitsposition erreicht werden soll. Diese Punkte gliedern sich wie folgt (siehe auch Abbildung 4.2) [52, S. viii]:

I: Governance und Sicherheit (orig. *Governance and Security*)

II: Management von Cloud-Identitäten (orig. *Managing Cloud Identities*)

III: Azure Netzwerksicherheit (orig. *Azure Network Security*)

IV: Azure KeyVault

V: Datensicherheit (orig. *Data Security*)

VI: Azure Security Center

VII: Azure Sentinel

VIII: Security Best Practices

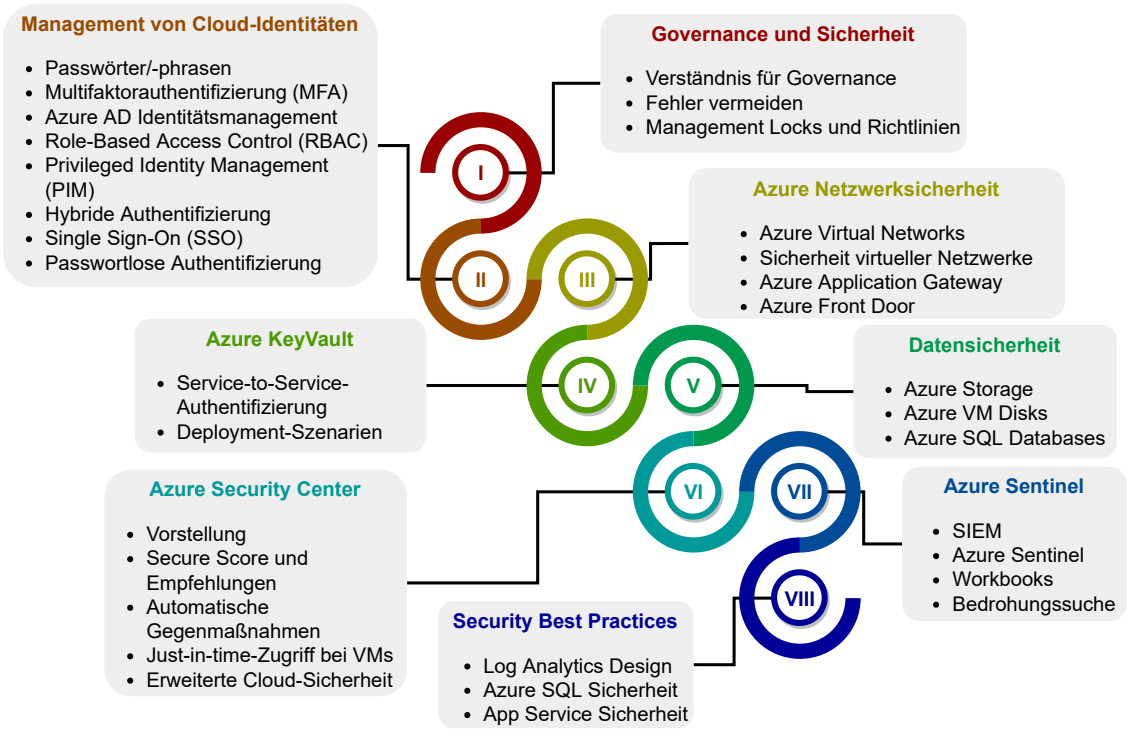

Abbildung 4.2: Ablauf und Inhaltspunkte von Mastering Azure Security [eigene Abbildung nach [52, S. 17, 48, 101, 127, 147, 169, 195, 213]]

Gemäß diesem Vorgehen steht am Anfang eine akkurate Planung und Realisierung von Richtlinien und Standards, denen ein *Azure*-Mandant und jegliche untergeordneten Komponenten zu folgen haben und welche stets strikt durchzusetzen sind. Darüber hinaus befasst sich diese Position mit der strukturellen Organisation von Abonnements und Services entsprechend unterschiedlicher Aufteilungen, welche die Organisationsstruktur der besitzenden Organisation auch in der Cloud widerspiegeln. Aufbauend auf einer funktionierenden *Governance* (engl. Steuerung) sind differenzierte Sicherheitsmaßnahmen zu realisieren, welche die einzelnen Bereiche einer *Azure*-Infrastruktur ansprechen und auf diese zugeschnitten sind. Hierbei wird herausgestellt, dass Identitäten, Rollenzuweisungen und klare Zugriffskontrolle der zentrale Aspekt einer funktionierenden Sicherheitsarchitektur in *Azure* darstellen [52, S. 18-19, 47-48].

Weiterhin gilt die Netzwerksicherheit als wichtiges Herzstück der Sicherheit in der Cloud, indem Ressourcen und Services für jegliche Angreifer unerreichbar gehalten werden. Hierzu können interne *Azure*-Dienste, aber auch individuelle Werkzeuge genutzt werden. Aufbauend darauf wird das *Azure KeyVault* betrachtet, welches den Schutz von Schlüsseln und Zertifikaten jeglicher Art gewährleistet. Zusätzlich referenziert dieser Punkt den sicheren Einsatz von *Infrastructure as Code* für das Ressourcen-Deployment. Nachfolgend ist der Datensicherheit Rechnung zu tragen, indem vor allem mittels Verschlüsselung eine weitere Sicherheitsebene hinzugefügt wird. Abgesehen hiervon sieht dieses Kapitel eine Klassifizierung von vertraulichen Daten vor [52, S. viii, 101-102, 127-128, 147].

Die Nutzung des *Azure Security Center* (ASC) wird für die Erkennung von Bedrohungen sowie die Bewertung der gesamten Sicherheit des betreffenden *Azure*-Mandanten empfohlen. Zudem wird vorgeschlagen, mithilfe der Empfehlungen des ASCs diese Sicherheitsposition stetig zu optimieren. In Kombination mit dem ASC wird die Nutzung des *Azure Sentinel*-Service beleuchtet, um mithilfe künstlicher Intelligenz Bedrohungen präventiv zu erkennen und bekämpfen sowie umfassend zu analysieren. Abgeschlossen wird Toromans Konzept von einer Reihe empfohlener *Best Practices* [52, S. 169-170, 195-197].

Azure Security Benchmark (ASB)

Dieser *Benchmark* wird von *Microsoft* selbst entwickelt und veröffentlicht und kommt integriert in das ASC zum Einsatz. Er dient der Ermöglichung schneller Entscheidungen bezüglich vielfältiger Basis-Sicherheitseinstellungen, indem anhand der Kontrollen des *Benchmarks* Empfehlungen für potenzielle Sicherheitsrisiken und deren Verbesserung ausgegeben werden. Hier wird darauf Wert gelegt, spezifisch solche Hinweise zu geben, welche nach Ansicht von *Microsoft* einen größtmöglichen positiven Effekt für den jeweiligen Service oder die Komponente mit sich bringen. Der *Benchmark* besteht sowohl aus so genannten *Security controls* als auch aus *Service baselines*. Der Unterschied zwischen diesen beiden Optionen wird nachfolgend in Tabelle 4.3 erläutert [19].

Tabelle 4.3: Differenzierung von Security controls und Service baselines [eigene Tabelle nach [19]]

Inhaltsart	**Erklärung**
Security controls	Empfehlungen, welche im Grunde für den gesamten *Azure*-Mandanten und somit jegliche Services anwendbar sind
Service baselines	Hierbei werden Kontrollen nur auf bestimmte Services angewandt, um Empfehlungen basierend auf der Sicherheit des betreffenden Dienstes anzubieten

Der ASB wird für neue Kunden, die ihre Sicherheit von Anfang an optimieren wollen, genauso wie für bereits bestehende *Azure*-Deployments empfohlen, mit dem speziellen Augenmerk auf die Minimierung von Risiken und hochrangigen Bedrohungen. Aber auch für Personen, welche lediglich *Azure*-Services und -Infrastrukturen auditieren, um deren Einklang mit gängigen Sicherheitsrichtlinien zu überprüfen, ist dieser *Benchmark* geeignet. Dies hat den Grund, dass sich die Kontrollen des ASB auf die Standards des CIS und des NIST abbilden lassen. Im Detail handelt es sich hierbei um die CIS-Kontrollen in der Version 7.1 sowie den NIST-Standard SP 800-53 (*Special Publication*). Folgende Kategorien beschreibt der ASB hierbei (siehe Abbildungen 4.3 und 4.4) [33]:

NS - Netzwerksicherheit (orig. *Network Security*)

IM - Identitätsmanagement (orig. *Identity Management*)

PA - Privilegierter Zugriff (orig. *Privileged Access*)

DP - Datenschutz (orig. *Data Protection*)

AM - Asset Management

LT - Logging und Bedrohungserkennung (orig. *Logging and Threat Detection*)

IR - Incident Response

PV - Sicherheits- und Schwachstellenmanagement (orig. *Posture and Vulnerability Management*)

ES - Endpunkt-Sicherheit (orig. *Endpoint Security*)

BR - Sicherung und Wiederherstellung (orig. *Backup and Recovery*)

GS - Governance und Strategie (orig. *Governance and Strategy*)

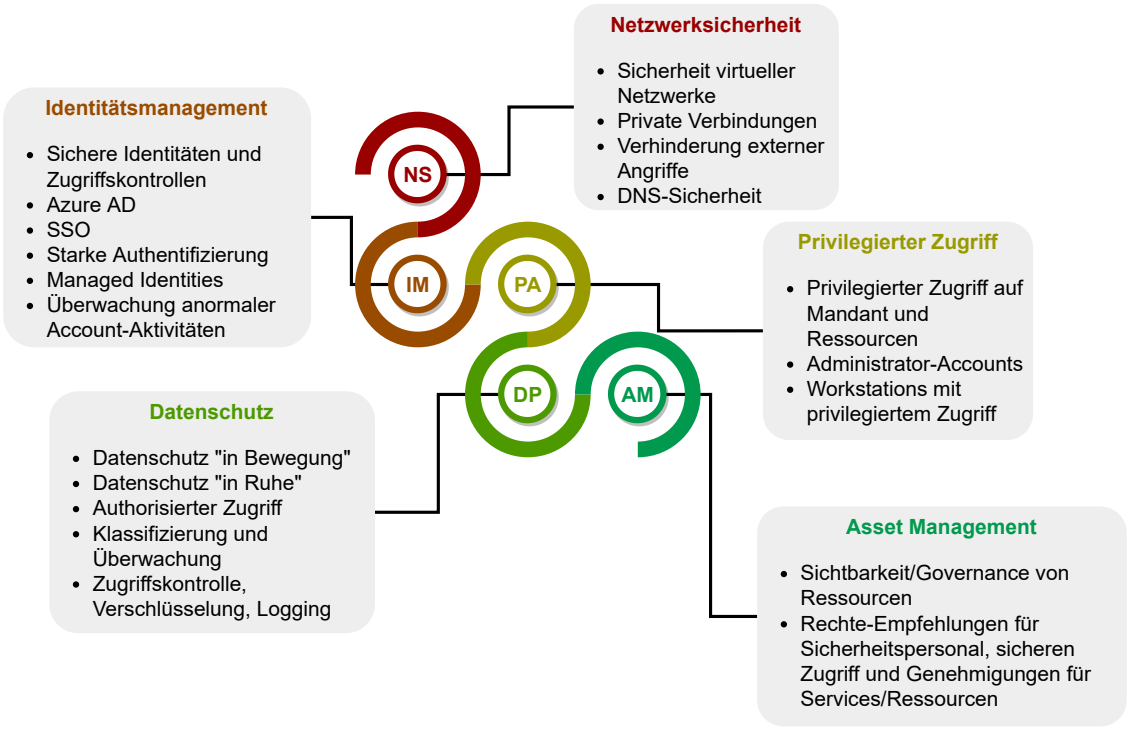

Abbildung 4.3: Übersicht und Ablauf des ASB, Teil 1 [eigene Abbildung nach [33]]

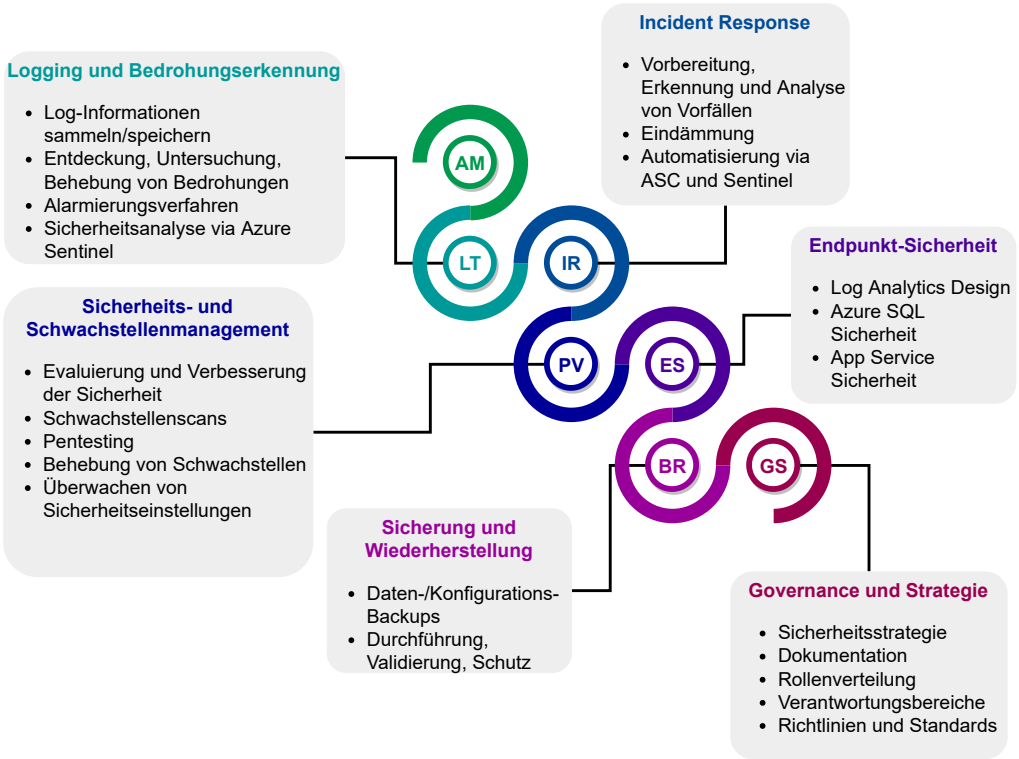

Abbildung 4.4: Übersicht und Ablauf des ASB, Teil 2 [eigene Abbildung nach [33]]

Jede Kontrolle des *Benchmarks* ist nach einem homogenen Schema aufgebaut. Diese enthält neben der Kennung der Kontrolle im ASB die Kennungen korrespondierender Kontrollen in den Standards des CIS und NIST. Darüber hinaus werden verschiedene wichtige Informationen preisgegeben, wie folgendes Beispiel zeigt[5]:

Azure ID: NS-1
CIS Controls v7.1 ID(s): 9.2, 9.4, 14.1, 14.2, 14.3
NIST SP 800-53 r4 ID(s): AC-4, CA-3, SC-7
Details: Ensure that all Azure virtual networks follow an enterprise segmentation principle that aligns to the business risks. Any system that could incur higher risk for the organization should be isolated within its own virtual network and sufficiently secured with either a network security group (NSG) and/or Azure Firewall.

Based on your applications and enterprise segmentation strategy, restrict or allow traffic between internal resources based on network security group rules. For specific well-defined applications (such as a 3-tier app), this can be a highly secure „deny by default, permit by exception“ approach. This might not scale well if you have many applications and endpoints interacting with each other. You can also use Azure Firewall in circumstances where central management is required over a large number of enterprise segments or spokes (in a hub/spoke topology).

Use Azure Security Center Adaptive Network Hardening to recommend network security group configurations that limit ports and source IPs based on external network traffic rules.

Use Azure Sentinel to discover the use of legacy insecure protocols such as SSL/TLSv1, SMBv1, LM/NTLMv1, wDigest, Unsigned LDAP Binds, and weak ciphers in Kerberos.
Responsibility: Customer
Customer Security Stakeholders:

- Security architecture
- Posture management
- Application Security and DevOps

[5] O.A.: **Security Controls V2: Network Security**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Azure*, 2021, Zugriff 05.09.2021, URL `https://docs.microsoft.com/en-us/security/benchmark/azure/security-controls-v2-network-security`

Die Abbildung der ASB-Kontrollen auf CIS- und NIST-Kontrollen ist jedoch mit Vorsicht zu betrachten. Dieser Umstand bedeutet lediglich, dass ein bestimmtes *Azure*-Feature genutzt werden kann, um ganz oder teilweise eine Kontrollanforderung des NIST oder CIS zu berücksichtigen. Die Nutzung dieses Features ist jedoch nicht gleichbedeutend mit der vollständigen Einhaltung der Kontrollen des CIS und NIST [33].

CIS Microsoft Azure Foundations Benchmark

Der *Microsoft Azure Foundations Benchmark* wird von der CIS publiziert, um eine Guideline zu bieten, nach der eine sichere Grundkonfiguration von *Azure*-Mandanten erfolgen kann. Spezifisch ist dieser *Benchmark* darauf ausgerichtet, diejenigen bei der Einrichtung eines stabilen Sicherheitsniveaus zu unterstützen, die gerade erst angefangen haben, *Microsoft Azure* zu adaptieren [23].

Verfasst werden die Dokumente des CIS stets basierend auf einem Konsens aus themenkundigen Experten. In diesem Fall handelt es sich um erfahrene Personen mit unterschiedlichen Hintergründen im Feld der IT, von Consulting über Softwareentwicklung und IT-Sicherheit bis hin zu Auditierung und Prüfung von Einhaltung anerkannter Standards der Daten- und Informationssicherheit. Der endgültige *Benchmark* entsteht hierbei in zwei Phasen der Konsensfindung [23].

Gleichzeitig ist diese Richtlinie keine allumfassende Darstellung sämtlicher sicherheitsbezogener Konfigurationsmöglichkeiten und Architekturelemente. Vielmehr soll dieser *Benchmark* als Grundlage dienen, auf welchem aufbauend spezifische, bedarfsgerechte Änderungen wann immer notwendig durchgeführt werden sollen. Die einzelnen Empfehlungen des Dokuments verfügen über einen so genannten Bewertungsstatus. Dieser gibt Aufschluss darüber, ob eine Empfehlung in ihrer Verifikation automatisiert werden kann oder manuelle Schritte erforderlich sind. Der Status trägt dementsprechend entweder den Wert **Automatisch** (orig. **Automated**) oder **Manuell** (orig. **Manual**). Neben dieser Einteilung existiert eine zweite Unterscheidung der Empfehlungen in **Level 1** und **Level 2**. Diese Unterscheidung wird in der folgenden Tabelle 4.4 erklärt [23].

Tabelle 4.4: Differenzierung zwischen Level 1- und Level 2-Empfehlungen des CIS [eigene Tabelle nach [23]]

Level	Erklärung
Eins (1)	**Ausrichtung:** praktisch und vorsichtig **Fokus:** Sicherheit und auf optimale Härtung ausgelegtes *Best Practice* **Auswirkungen:** Begrenzung des potenziellen Einflusses einer Technologie auf ein akzeptables Maß

Fortsetzung von Tabelle 4.4	
Zwei (2)	**Ausrichtung:** Umgebungen oder Anwendungsfälle mit erhöhtem Sicherheitsbedarf, über die Erhaltung von komfortablem Management und Benutzung hinaus **Fokus:** *Defense in Depth* **Auswirkungen:** potenzielle Einschränkung der Leistung/des Nutzens der betreffenden Technologie **Zusatz:** mögliche zusätzliche Lizenzkosten oder Einsatz von Drittanbietersoftware erforderlich

Wie *Mastering Azure Security* und der ASB weist auch der CIS *Benchmark* bestimmte Fokuspunkte auf, welche mit den enthaltenen Kontrollen und Empfehlungen angesprochen werden. Diese gliedern sich wie folgt (siehe Abbildung 4.5) [23]:

1: Identitäts- und Zugriffsmanagement (orig. *Identity and Access Management*)
2: Security Center
3: Speicherkonten (orig. *Storage Accounts*)
4: Datenbank-Services (orig. *Database Services*)
5: Logging und Überwachung (orig. *Logging and Monitoring*)
6: Netzwerke (orig. *Networking*)
7: Virtuelle Maschinen (orig. *Virtual Machines*)
8: Weitere Sicherheitserwägungen (orig. *Other Security Considerations*)
9: AppService

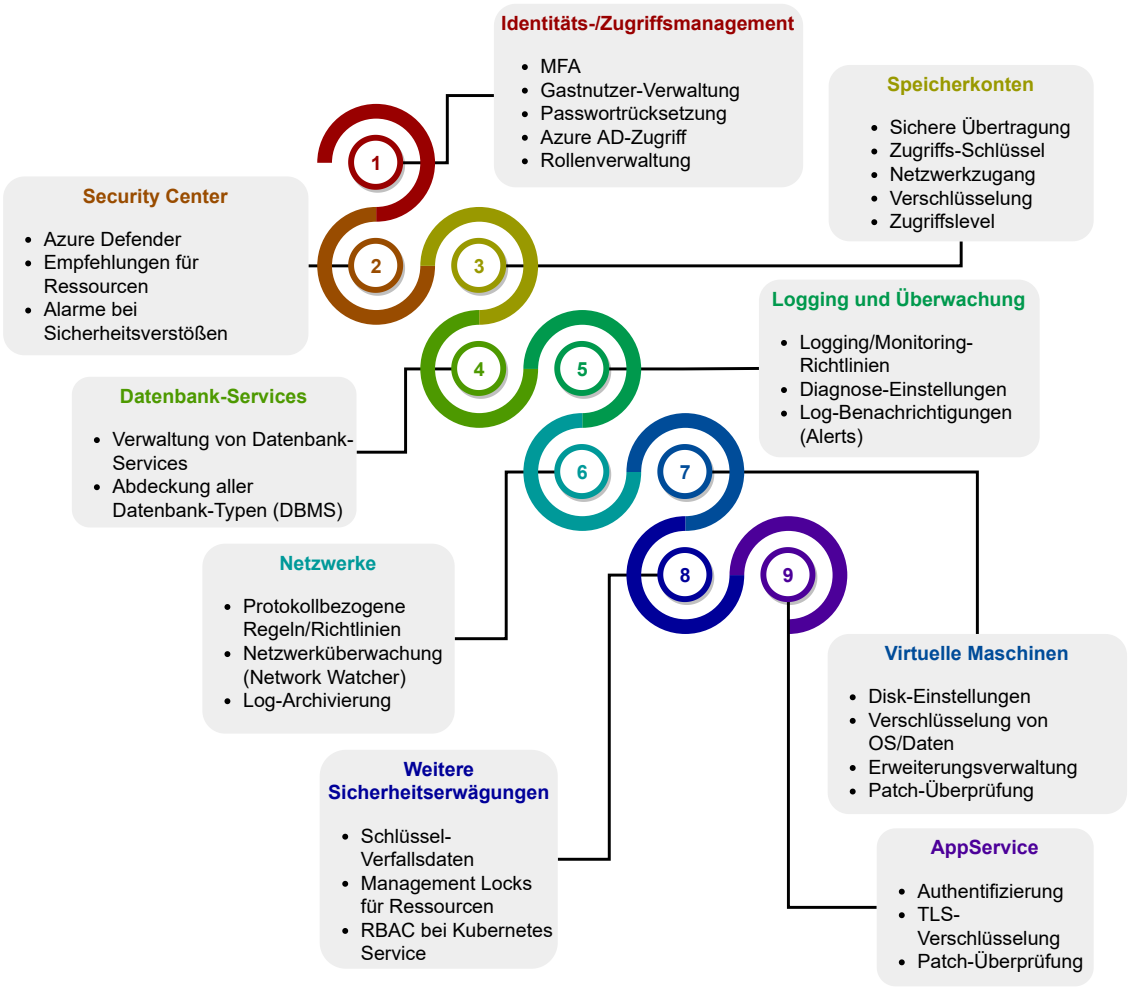

Abbildung 4.5: Übersicht und Ablauf des CIS Benchmarks für Azure [eigene Abbildung nach [23]]

Eine exemplarische Empfehlung, wie sie im CIS *Benchmark* auftritt, befindet sich in Anhang A im Abschnitt CIS Benchmark-Beispiel (siehe Seite 99), um den zugrundeliegenden Aufbau und Inhalt zu verdeutlichen.

4.2.2 Einsatzempfehlung

Ausgehend von den beschriebenen Vorgehensweisen für das *Hardening* in *Azure* ergeben sich eine Reihe Auswahlkriterien, anhand derer der Einsatz der einzelnen Methoden für verschiedene Szenarien empfohlen werden kann. Diese sind **Technische Tiefe**, **Organisatorische Tiefe**, **Nutzung nativer Technologien**, **Industrieller Standard**, erreichbares **Sicherheitsniveau** sowie **Automatisierbarkeit**. Die Tabelle 4.5 ordnet die einzelnen Kriterien den Vorgehensweisen zu.

Mastering Azure Security bietet aufgrund der enthaltenen detaillierten technischen Beschreibungen der einzelnen Schritte sowie den ebenso ausgeprägten Hintergrunderklärungen eine große technische Tiefe. Daneben wird auch die organisatorische Seite eines *Hardening Checks* bedacht, indem umfassende Erklärungen und Anweisungen zu den Themen *Governance*, Aufteilung der Ressourcen eines Mandanten und Richtlinienerstellung und -management integriert werden. Des Weiteren erläutern Toroman und Janetscheck die Nutzung der nativen *Azure*-Sicherheitstechnologien, beispielsweise ASC und *Azure Sentinel*. Hierbei wird ein solides Grund-Sicherheitsniveau für den Einstieg in *Azure* oder die Absicherung eines bestehenden *Azure*-Mandanten als Ziel beschrieben. Die Konfigurationsmaßnahmen, welche Toroman und Janetscheck hierbei in ihre Vorgehensweise einbeziehen, werden vollständig manuell durchgeführt beziehungsweise überprüft. Das festgelegte Ziel liegt darin, einen Anhaltspunkt für den Entwurf und die Aufrechterhaltung sicherer Infrastrukturen in *Azure* zu bieten.

Zu empfehlen ist *Mastering Azure Security*, wenn hohe technische und organisatorische Tiefe bei einem *Hardening Check* erforderlich sind, beispielsweise bei der Einrichtung oder Evaluation einer weitreichenden Infrastruktur, gegebenenfalls inklusive einer Überprüfung der Aufteilung eines *Azure*-Mandanten und einer angemessenen Umstrukturierung. Dies bedeutet eine besondere Eignung dieses Vorgehens für den Einsatz bei IaaS sowie für das tiefgreifende *Hardening* eines gesamten *Azure*-Mandanten.

Der ASB fokussiert sich auf die wesentlichen Aspekte eines *Hardening Checks*, indem akkurate Empfehlungen zu einzelnen Konfigurationseinstellungen oder wichtigen Sicherheitsaspekten gegeben werden. Hierbei werden zwar weiterführende Informationen verlinkt, jedoch nicht in die *Hardening*-Vorgehensweise miteinbezogen, wodurch keine derartige technische Tiefe entsteht. Gleichzeitig wird der organisatorische Aspekt des *Hardening*, *Governance*, mit verschiedenen Empfehlungen umrissen, jedoch eher an der Obefläche, da diese Seite der IT-Sicherheit prinzipiell in den Aufgabenbereich des Kunden fällt. Native Technologien werden direkt in den *Hardening*-Prozess als Empfehlungen in verschiedenen Aspekten miteinbezogen, insbesondere das ASC. Zudem beziehen sich die Kontrollen des ASB auf inhaltlich ähnliche beziehungsweise gleiche Empfehlungen bekannter Industriestandards des NIST und CIS. Allerdings strebt auch der *Benchmark* von *Microsoft* lediglich ein stabiles Grund-Sicherheitsniveau für den Einstieg in *Azure* beziehungsweise die von Anfang an garantierte Überprüfung der Sicherheit eines *Azure*-Mandanten an. Ausgehend von der Realisierung vieler Empfehlungen über native Technologien lässt sich diese Vorgehensweise teilweise automatisieren.

Zu empfehlen ist der ASB für einen allgemeinen *Hardening Check*, welcher an gültige Industriestandards anlehnt. Der Fokus liegt in diesem Fall auf einer schnellen und nativen Realisierung eines stabilen Sicherheitsniveaus. Eine Eignung für ein spezifisches Service-Modell lässt sich nicht feststellen, vielmehr ist der ASB am besten immer über einen gesamten *Azure*-Mandanten einzusetzen.

Der *Benchmark* des CIS verfolgt ein ähnliches Ziel wie der ASB. Die technische Tiefe wird durch eine umfangreiche Bandbreite von Sicherheitsempfehlungen gewährleistet, welche mit detaillierten Beschreibungen, Begründungen sowie Durchführungsanleitungen versehen sind. Die organisatorische Tiefe andererseits kommt in dieser Vorgehensweise nicht zur Sprache. Ebenso legt das CIS keinen Fokus auf etwaige native Sicherheitslösungen in *Azure*. Es handelt sich jedoch um einen industriellen Standard, welcher auf einem breiten Konsens aus Experten der IT-Sicherheit basiert und stets aktualisiert wird. Zudem existiert eine Vielzahl Empfehlungen, welche über ein solides Grund-Sicherheitsniveau auch für ein erweitertes Schutzniveau für Systeme mit erhöhtem Sicherheitsbedarf eingesetzt werden können. Ähnlich dem ASB ist der *Benchmark* des CIS in Teilen bei der Durchführung der Kontrollen im Rahmen eines *Hardening Checks* automatisierbar.

Zu empfehlen ist der *CIS Microsoft Azure Foundations Benchmark*, wenn bei einem *Hardening Check* eine umfassende technische Überprüfung anhand eines Industriestandards vorgenommen werden soll, die auch teilautomatisiert ablaufen kann. Zudem können Objekte, welche einen erhöhten Schutzbedarf aufweisen, ebenso mit den Empfehlungen des CIS gehärtet werden. Ähnlich dem ASB existiert hier keine Ausrichtung auf ein spezifisches Service-Modell, sondern ein Fokus auf eine ganzheitliche Sicherung eines *Azure*-Mandanten, um universell einsetzbar zu sein.

Tabelle 4.5: Bewertung und Empfehlung der Hardening-Vorgehensweisen nach zentralen Kriterien [eigene Tabelle]

		Vorgehensweisen		
		Mastering Azure Security	ASB	CIS Benchmark
Kriterien	Technische Tiefe	✓		✓
	Organisatorische Tiefe	✓	(✓)	
	Nutzung nativer Technologien	✓	✓	
	Industrieller Standard		(✓)	✓
	Sicherheitsniveau	Grund	Grund	Hoch
	Automatisierbarkeit		(✓)	(✓)

✓	Erfüllt das Kriterium
(✓)	Erfüllt das Kriterium teilweise
Grund	Grundlegendes Sicherheitsniveau
Hoch	Hohes Sicherheitsniveau

4.2.3 Zu nutzende Tools

Für das *Hardening* als Sicherheitstest bei *Azure* existieren eine Reihe von Tools, welche für das Überprüfen und Sicherstellen der Vielzahl von Empfehlungen eines *Benchmarks* voll- oder teilautomatisierte Unterstützung bieten können. Insbesondere *Azure* selbst stellt unterschiedliche Services bereit, welche verwendet werden können, um die Sicherheit eines *Azure*-Mandanten zu überwachen, zu verifizieren und zu verbessern [9, S. 119].

Azure Security Center

Das *Azure Security Center* ist die zentrale Überwachungsmöglichkeit der Sicherheit und Gesundheit jeglicher im Einsatz befindlicher Ressourcen. Zudem wird ein täglicher Abgleich mit den Kontrollen und Empfehlungen des ASB durchgeführt, welcher somit auch dazu beiträgt, zu treffende Konfigurationseinstellungen, beispielsweise bei der Nutzerverwaltung, Ressourcenorganisation etc. im Blick zu behalten. Hauptziel des *Security Center* ist der andauernde Schutz eines *Azure*-Mandanten vor sich schnell entwickelnden Bedrohungen. Darüber hinaus wird eine Sicherstellung der Befolgung von *Best Practices* beim Ressourcen-Deployment angestrebt [9, S. 119-120]. Die Hauptseite des *Security Centers* ist in Abbildung 4.6 zu sehen.

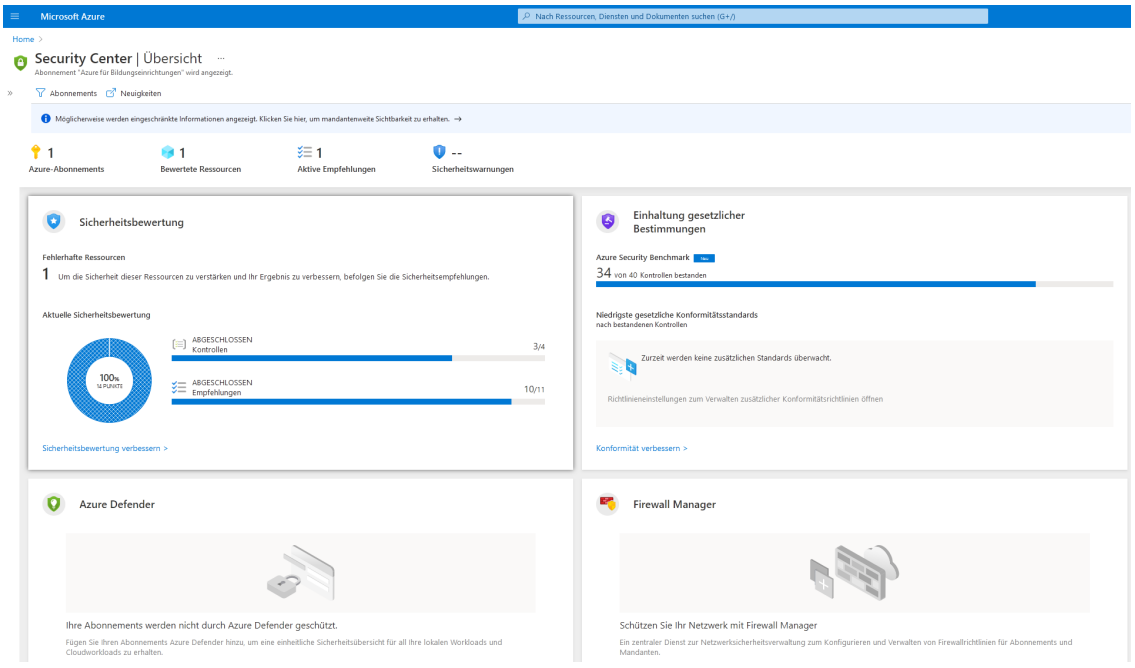

Abbildung 4.6: Übersichtsseite des Azure Security Center [Screenshot]

Neue Abonnements oder Ressourcen werden automatisch in die Überprüfung zugewiesener Richtlinien mit einbezogen. Hierbei werden gemäß des ASB beispielsweise Parameter wie Betriebssystemversionen oder die Installation und Aktivierung eines *Log Analytics Agent* verifiziert. Die kontinuierliche Prüfung der *Azure*-Infrastruktur wird mit einem *Security Score* auf einer Skala von null bis 100 Prozent anschaulich dargestellt. Zudem existiert eine Übersicht darüber, welche Empfehlungen erfüllt werden sowie eine Reihe von Hinweisen, wie nicht erfüllte Empfehlungen zu erreichen sind. Hierbei besteht teilweise die Möglichkeit, den Fix direkt aus dem *Security Center* heraus anzuwenden [9, S. 120-122].

Zuletzt erfasst das *Azure Security Center* anormale Aktivitäten in einzelnen Ressourcen, um Aufschluss über potenzielle Bedrohungen zu geben. Hierbei wird eine Aktivität mit Informationen zum Startzeitpunkt, betroffenen Ressourcen sowie korrespondierenden MITRE ATT&CK-Taktiken ausgebaut. Zusätzlich stellt das *Security Center* Informationen zur Behebung oder Minimierung einzelner Schwachstellen bereit. Dieser Prozess kann über so genannte logische Apps bei Bedarf automatisiert werden [9, S. 122-123].

Azure Sentinel

Bei *Azure Sentinel* handelt es sich um das integrierte *Security Information Event Management*- und *Security Orchestration Automated Response*-Tool (SIEM, SOAR). Mithilfe dieses Services ist eine Verarbeitung der Log-Dateien einer Vielzahl von *Azure*-Services, inklusive VMs, möglich, um dadurch Aufschluss über potenzielle Bedrohungen zu geben und deren Analyse zu unterstützen [9, S. 124]. Die Übersichtsansicht des *Azure Sentinel* wird in Abbildung 4.7 dargestellt.

Hierdurch werden Bedrohungen und anormale Aktivitäten durchgehend sichtbar gemacht und das Verfolgen basierend auf dem MITRE-Framework präventiv ermöglicht. Jedes registrierte Event kann näher betrachtet werden, um Aufschluss über den genauen Zeitpunkt, den betroffenen Account sowie den Ausgangspunkt der Meldung zu geben. Zudem bietet *Azure Sentinel* vielfältige so genannte *Hunting queries*, mit deren Hilfe Events in Beziehung zueinander gesetzt werden können [9, S. 124].

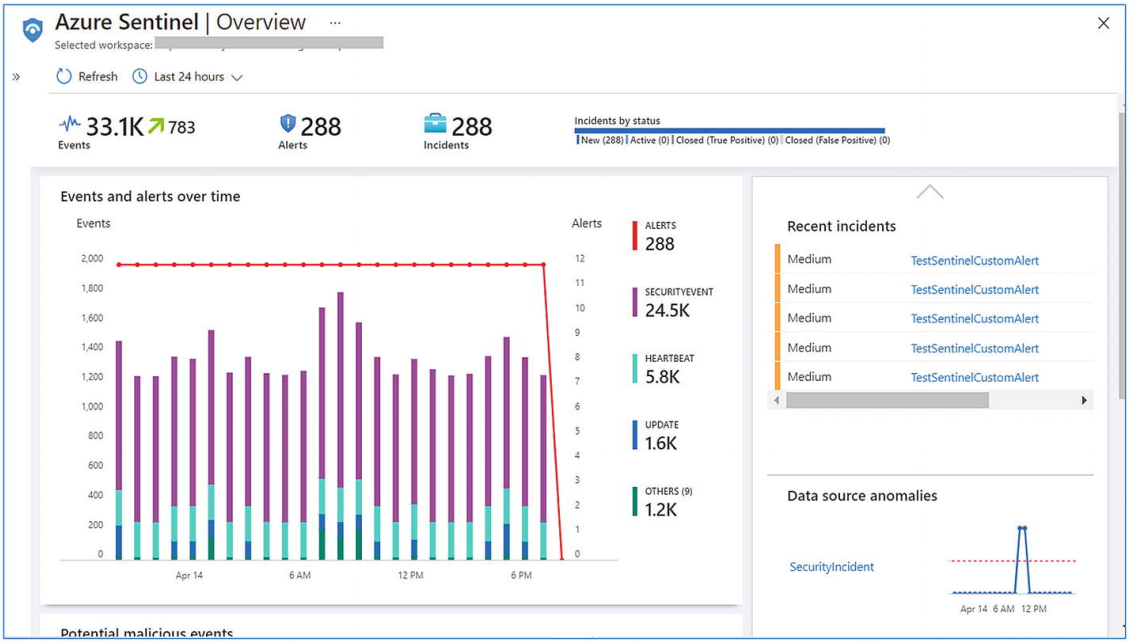

Abbildung 4.7: Übersicht des Azure Sentinel [Abbildung aus [9, S. 125]]

Azure Defender

In Kombination mit dem *Azure Security Center* und dem *Azure Sentinel* bietet der *Azure Defender* erweiterte Sicherheit und Bedrohungserkennung für Windows- und Linux-VMs sowie verschiedene andere *Azure*-Services, beispielsweise DNS. Erkannte Bedrohungen werden im *Security Center* gemeldet und dargestellt. Dies erlaubt tiefergehende Analysen und Schritte zur Behebung der Bedrohung. Werden Windows-Server in den Überwachungsrahmen des *Security Centers* aufgenommen, wird der *Defender* automatisch hinzugezogen. Die Vielzahl von Leistungen dieses Services erstreckt sich über die Integration von *Log Analytics Agents*, dateilose Angriffserkennung, *Docker Hardening*, adaptives Netzwerk-Hardening, *Just-In-Time*-VM-Zugriff (JIT), Dateiverlaufsüberwachung sowie erweiterte Sicherheitskontrollen [9, S. 126].

Nessus

Der Schwachstellenscanner *Nessus* der Firma *Tenable* bietet ebenfalls die Möglichkeit, eine *Azure*-Infrastruktur zu auditieren, mit Hinblick auf Fehlkonfigurationen innerhalb der Cloud-Umgebung sowie in den Accounteinstellungen. [17] Hierbei nutzt das spezifische *Nessus-Plugin* die *Azure REST API* (*Representational State Transfer*), um die einzelnen Konfigurationen abzufragen. Diese API antwortet mit JSON (*JavaScript Object Notation*) [31].

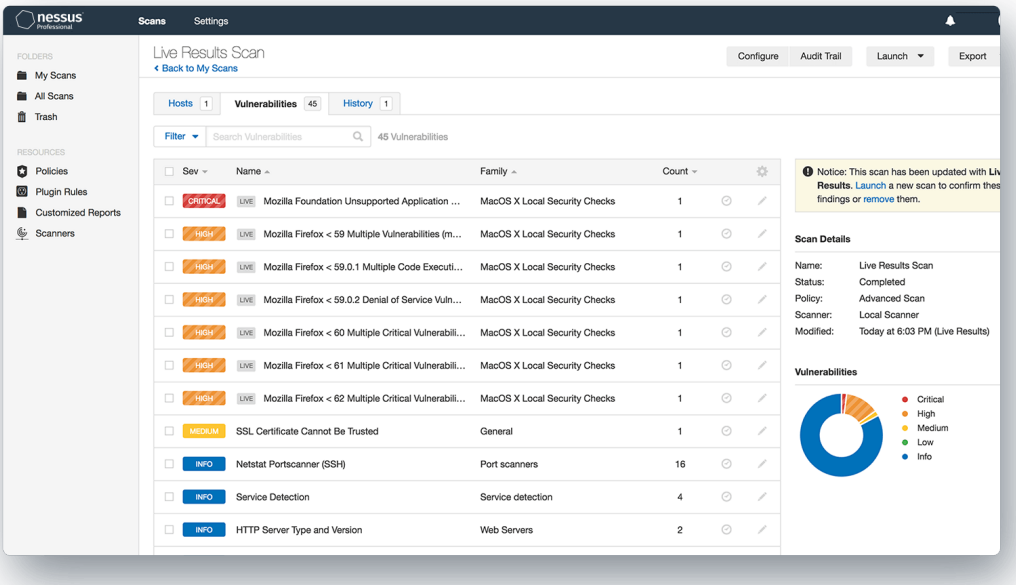

Abbildung 4.8: Oberfläche des Schwachstellenscanners Nessus [Abbildung von [32]]

4.2.4 Best Practices

Microsoft

Microsoft selbst hat eine Reihe von *Security Best Practices* für *Azure* veröffentlicht, genannt *Azure security top 10*. Diese erstrecken sich über die vier Kategorien Personen (orig. *People*), Prozess (orig. *Process*), Technologie (orig. *Technology*) und Architektur (orig. *Architecture*). Laut dem *Cloud Provider* basieren diese auf „gelernten Lektionen“, sprich realen Situationen, die von Kunden wie auch dem *Provider* selbst erfahren wurden. Jede Praktik besteht mindestens aus den Angaben Was, Warum, Wer und Wie [20]. Die folgende Tabelle 4.6 bietet eine kurze Übersicht über die von *Microsoft* präsentierten Punkte.

Tabelle 4.6: Best Practices für Azure-Sicherheit der Firma Microsoft [eigene Tabelle nach [20]]

Kategorie	**Titel**	**Kurzbeschreibung**
Personen	*1. Educate teams about the cloud security journey*	Die Migration in die Cloud bringt neue Bedrohungen und Angriffsvektoren mit sich. Es wird empfohlen, Personal, welches mit Informationssicherheit betraut ist, entsprechend zu schulen und dahingehendes Wissen zu überprüfen
	2. Educate teams on cloud security technology	Gepaart mit neuen Sicherheitsrisiken bringt die Cloud auch neue Technologien mit sich, ein angemessenes Sicherheitsniveau bereitzustellen. Personal, welches sich um die IT-Sicherheit in *Azure* kümmert, sollte sich über diese Technologien informieren und diese verstehen
Prozess	*3. Assign accountability for cloud security decisions*	Für Entscheidungen, welche die Sicherheit in *Azure* betreffen, sollten Personen ernannt werden, welche die notwendigen Kompetenzen erhalten, diese zu treffen und die entsprechende Verantwortung zu tragen
	4. Update incident response processes for cloud	Entsprechende *Incident Response Teams* sollten darin unterwiesen werden, wie auf Sicherheitsvorfälle zu reagieren ist. Darüber hinaus sollten diese sich mit potenziellen nativen hierbei zu verwendenden Tools und dem Umgang mit diesen vertraut machen. Ebenfalls sollten Simulationen für derartige Vorfälle durchgeführt werden

Fortsetzung von Tabelle 4.6

	5. Establish security posture management	Zwar ist das Einsetzen von Diensten in *Azure* in den meisten Fällen eine Frage weniger Klicks, dennoch ist es erforderlich, deren sichere Konfiguration und Überwachung auf etwaige Sicherheitsrisiken sicherzustellen. Hierbei ist zu überprüfen, ob Services gemessen daran, wofür sie eingesetzt werden, ausreichend gesichert sind
Technologie	*6. Require passwordless or multifactor authentication*	Für eine erhöhte Sicherheit sollten besonders privilegierte Nutzerkonten alternative passwortlose Anmeldeverfahren (z.B. Biometrie) oder MFA als zweite Sicherheitsstufe nutzen
	7. Integrate native firewall and network security	Jegliche Netzwerkaspekte sollten durch native Lösungen, beispielsweise *Azure (Web App) Firewall* und (D)DoS-Schutz gesichert werden. Zwar sind hier auch individuelle Lösungen denkbar, jedoch können diese komplex in ihrer Umsetzung werden. Deshalb sollte auf integrierte Lösungen zurückgegriffen werden
	8. Integrate native threat detection	Ähnlich wie bei der vorherigen Empfehlung sollten native Technologien für Bedrohungserkennung und SIEM verwendet werden. Dies hat den Grund, dass herkömmliche Werkzeuge in diesem Bereich für den *On Premise*-Einsatz entwickelt wurden, eine Cloud-Infrastruktur jedoch andere Ansätze erfordert. Zudem ist hier Kompatibilität vonseiten des *Providers* gegeben

Fortsetzung von Tabelle 4.6

Architektur	*9. Standardize on a single directory and identity*	Der Einsatz eines einzigen *Azure AD* sowie einzigartiger Accounts für Benutzer und Services verhindert die Wiederverwendung von gleichen Passwörtern und minimiert die Existenz verwaister Konten, Sicherheitsrisiken, welche im Falle eines Angriffs ausgenutzt werden könnten. Diese Praxis ist stets zu verifizieren
	10. Use identity-based access control (instead of keys)	Schlüsselbasierte Authentifizierung kann genutzt werden, um sich bei Services und APIs zu authentifizieren, ist jedoch mit zunehmender Größe schwer zu verwalten. Identitätsbasierte Authentifizierung bietet hier Abhilfe
	11. Establish a single unified security strategy	Eine inkohärente Sicherheitsstrategie kann zu Wechselwirkungen verschiedener parallel ablaufender Arbeitsprozesse führen, welche letzten Endes ein Sicherheitsrisiko darstellen. Wird beispielsweise die Netzwerksicherheit mit einer anderen Prämisse als das Identitätsmanagement entwickelt, besteht Potenzial für eine Vielzahl notwendiger Firewall-Ausnahmen, welche andernfalls hätten vermieden werden können

Mastering Azure Security

Toroman und Janetscheck empfehlen in *Mastering Azure Security* ebenfalls drei *Best Practices* für die Aufrechterhaltung der Sicherheit in *Azure*. Diese betreffen folgende Themen [52, S. 213]:

- *Log Analytics*-Design
- Sicherheit von *Azure* SQL-Datenbanken
- Sicherheit des *Azure App Service*

Log Analytics-Design besitzt deshalb eine große Wichtigkeit, weil die Performance von ASC und *Azure Sentinel* stark von diesem Service abhängig ist. Deshalb wird empfohlen, so wenig *Log Analytics*-Arbeitsbereiche wie möglich einzusetzen sowie diese nach Möglichkeit regional zu organisieren, um Bandbreitenkosten vorzubeugen. Ein einzelner Arbeitsbereich mindert den Organisationsaufwand bezüglich der Rechteverwaltung zwar, ist jedoch erhöhten Bandbreitenkosten gegenüberzustellen. Sind Services und Ressourcen in verschiedenen Regionen im Einsatz, ist daher eine regionale Verteilung des *Log Analytics*-Dienstes zu erwägen. In diesem Fall ist jedoch zu berücksichtigen, dass der entsprechende *Monitoring Agent* beispielsweise in einer VM manuell oder anderweitig automatisiert eingesetzt werden muss. Hier ließe sich auf *Infrastructure as Code* (IaC) zurückgreifen. [52, S. 213-215].

In Kombination mit den innerhalb der Vorgehensweise der Autoren thematisierten Sicherheitsmaßnahmen für Datenbanken (siehe Abbildung 4.2, **V: Datensicherheit**, Seite 52) wird explizit der Einsatz einer *Firewall* empfohlen. Somit kann beispielsweise jeglicher Zugriff von unautorisierten IP-Adressen direkt unterbunden werden. In Fällen, in denen eine einzige IP-Adresse Zugriff auf eine Datenbank erhalten soll, ist jedoch die Ressourcenplanung zu überdenken, da *Firewall*-Regeln auf der Ebene eines *Azure SQL Server* verwaltet werden und eine derartige Freigabe einen Zugriff von dieser IP auf jegliche Datenbanken auf demselben Server zur Folge hätte. Folglich sollten Datenbanken gemäß der gleichberechtigten Nutzung von Services oder Nutzern organisiert werden [52, S. 215-217].

Mithilfe des *Azure App Service* können innerhalb kurzer Zeit HTTP-basierte Web-Anwendungen und APIs gehostet werden, die eine Vielzahl von Programmiersprachen unterstützen. Hierbei sind eine große Anzahl Faktoren für die Sicherheit zu bedenken: Zugriff, Protokolle, Zertifikate und weitere essenzielle Eigenschaften. Bezüglich des Zugriffs und der Authentifizierung bietet eine Applikation, die mit dem *App Service* erstellt wurde, vielfältige Möglichkeiten inklusive Drittanbieterunterstützung für beispielsweise *Google* oder *Facebook*. Hier wird empfohlen, ausschließlich eine Authentifizierung über *Azure AD* zu ermöglichen, da hier eine native Lösung mit der größtmöglichen Kontrolle über weitere Sicherheitsvorkehrungen gewählt wird. Ebenfalls wichtig ist die Beschrän-

kung der zu nutzenden Protokolle der Web-Anwendung. Hier besteht die Möglichkeit, die Anwendung zu zwingen, ausschließlich HTTPS (*Hypertext Transfer Protocol Secure*) zu nutzen sowie Zertifikate zu setzen. Zuletzt sind die Netzwerkoptionen der Anwendung zu betrachten und der sichere Zugriff auf diese Ressource zu konfigurieren. Hier empfiehlt sich eine Integration in ein virtuelles Netzwerk. Darüber hinaus sollten strikte Zugriffsbeschränkungen implementiert sowie eine eigene Web-Applikations-Firewall (*Azure Front Door*) bereitgestellt werden [52, S. 217-226].

Schlüsselergebnisse, welche für das *Hardening* eines *Azure*-Mandanten aus *Mastering Azure Security* behalten werden sollten, sehen wie folgt aus [52, S. 226]:

- Identitäten und Geheimnisse (orig. *Secrets*) sind mit jeglichen verfügbaren Mitteln zu schützen
- Zugriff sollte möglichst entweder dem Muster *Just Enough Administration* (JEA) oder *Just-In-Time* Administration (JIT) folgen
- Lediglich spezifisch erforderliche Zugriffspunkte sollten öffentlich erreichbar sein. Services und Ressourcen, welche Management- und Sicherheitsaufgaben erfüllen, sollten jedoch niemals öffentlich erreichbar sein
- Sämtliche Daten sollten stets verschlüsselt gespeichert und transportiert werden
- Das ASC und *Azure Sentinel* sollten aktiviert und konfiguriert sein, da auf diese Weise Einblicke in die Sicherheit eines *Azure*-Mandanten gewonnen werden und schnellere Aktionen/Reaktionen bei Sicherheitsverstößen erfolgen können
- Neben dedizierten Services für die Sicherheit in *Azure* besitzt jeder Service eigene Sicherheitseinstellungen. Diese sollten ebenfalls genutzt werden

4.3 Azure-Penetrationstest

In Zusammenhang mit der Sicherheit von *Azure* existieren bekannte, traditionelle Angriffe wie XSS, (D)DoS oder *Brute Forcing* weiterhin, können jedoch durch die Beschaffenheit der *Azure Cloud* besser in ihrer Gefahr gemindert werden. Beispielsweise besteht die Möglichkeit für ein automatisches *Patch Management* bei PaaS-Anwendungen. Zudem ist die (physische) Basis-Infrastruktur einer Cloud bis zu einem gewissen Grad resilienter gegen Ausfälle, indem Lasten ausgleichend verteilt werden. Andere Angriffe rücken stärker in den Fokus, zum Beispiel Verluste des Datenschutzes oder der Privatsphäre. Dies hat mitunter den Grund, dass die Cloud weltumspannend verteilt ist und Daten zugunsten von Redundanz und Ausfallsicherheit dezentral gespeichert werden können. Zusätzlich bilden sich neue Risiken, wie *Privilege Escalation* von VM zu VM oder VM zu Host, *Jailbreaks* aus einer VM heraus oder *Hyperjacking*, ein Angriff, bei dem mittels eines Rootkits versucht wird, Einfluss auf einen Host oder eine andere VM zu nehmen [45, S. 7].

Ein Penetrationstest kann Aufschluss über bestehende Risiken und Gefahren für eine *Azure*-Infrastruktur geben. Hierbei muss ein Tester die Taktiken, Techniken und Prozeduren verstehen (orig. *Tactics, Techniques and Procedures* - TTP), mit deren Hilfe ein *Azure*-Mandant angegriffen werden kann. Dies dient der Identifikation von Schwachstellen und Sicherheitslücken, welche einem Angreifer ebenfalls in die Hände fallen könnten [3, S. xxii].

4.3.1 Bekannte Vorgehensweisen

Wie für das Azure Hardening existieren für Penetrationstests ebenfalls Vorgehensweisen, welche die speziellen Gegebenheiten der *Azure Cloud*-Umgebung berücksichtigen. Hierbei liegt der Fokus nicht auf der manuellen oder teilweise automatisierten Überprüfung von Konfigurationseinstellungen, um die Sicherheit eines *Azure*-Mandanten zu verifizieren, sondern auf der Entdeckung von Schwachstellen und deren Ausnutzung, ähnlich wie ein realer Angriff auf eine *Azure*-Infrastruktur ablaufen würde. Auch hier existieren Abläufe und zentrale Gesichtspunkte, welche berücksichtigt werden müssen. Nachfolgend werden zwei Vorgehensweisen für Penetrationstests bei *Microsoft Azure* erläutert. Dabei handelt es sich hierbei um das Buch **Pentesting Azure Applications** von Matt Burrough sowie um das **Cloud Penetration Testing Playbook** der CSA.

Pentesting Azure Applications

Ein Penetrationstest an einer *Azure*-Infrastruktur ist prinzipiell eine Untersuchung der Sicherheit einer oder mehrerer Abonnements. Dieser Umstand steht damit in Zusammenhang, dass die Hard- und Software, welche *Azure* als Plattform konstituieren, genannt *Azure Fabric*, *Microsoft* gehören und bei einem Penetrationstest somit stets außerhalb des Testumfangs liegen. Matt Burrough versucht daher in *Pentesting Azure Applications* eine Vorgehensweise für das Testen von *Azure*-Abonnements zu bieten, die aus Techniken besteht, welche sowohl angemessen als auch regulatorisch erlaubt sind [3, S. xxii-xxiii].

Inhaltlich gliedert Burrough den Ablauf eines *Azure*-Penetrationstests in acht zentrale Abschnitte. Jeder dieser Abschnitte beschreibt ein Aufgabenpaket, welches bei der Durchführung des Tests bedacht werden sollte. Als Gesamtbild ergibt sich somit eine grundlegende Reihenfolge von Schritten, nach der ein Penetrationstest bei *Azure* durchgeführt werden kann. Diese gliedert sich wie folgt (siehe Abbildung 4.9) [3, S. xxiii]:

Kapitel 1: Vorbereitung (orig. *Preparation*)

Kapitel 2: Zugriffsmethoden (orig. *Access Methods*)

Kapitel 3: Aufklärung (orig. *Reconnaisance*)

Kapitel 4: Speicher untersuchen (orig. *Examining Storage*)

Kapitel 5: VMs anvisieren (orig. *Targeting Virtual Machines*)

Kapitel 6: Netzwerke untersuchen (orig. *Investigating Networks*)

Kapitel 7: Andere *Azure*-Services (orig. *Other Azure Services*)

Kapitel 8: Überwachung, Logs und Benachrichtigungen/Alarme (orig. *Monitoring, Logs and Alerts*)

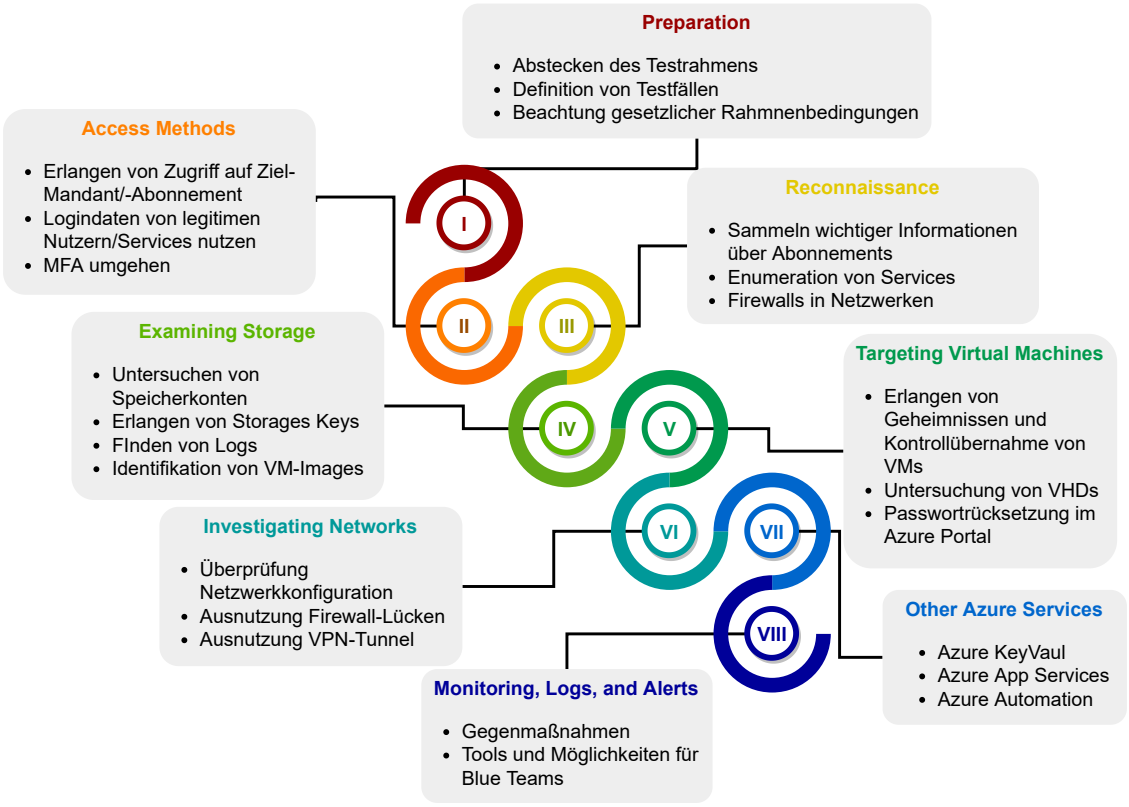

Abbildung 4.9: Ablauf und Inhaltspunkte von Pentesting Azure Applications [eigene Abbildung nach [3]]

Die Reihenfolge ist so aufgebaut, wie Burrough seine eigenen Tests in *Azure* durchführt. Hierbei ist darauf hinzuweisen, dass in den meisten Fällen nicht jedes Kapitel tatsächlich benötigt wird. Wie in herkömmlichen Penetrationstests auch, wo nicht stets dieselben Services und Technologien eingesetzt werden, nutzt nicht jeder Kunde dieselben *Azure*-Dienstleistungen, sondern vielmehr eine kleinere Gruppe von Leistungen, womit in vielen Fällen einzelne Punkte auszulassen sind [3, S. xxiii].

Begonnen wird mit der Planung des Testumfangs und Vorbereitung der detaillierten Rahmenbedingungen. In diesem Fall gewinnt dieser Prozess, welcher am Anfang eines jeden Penetrationstests steht, noch mehr an Bedeutung, da in der Cloud ansonsten klare Grenzen zwischen den Infrastrukturen verschiedener Kunden und auch zu Bereichen, welche dem *Provider* unterliegen, weicher und variabler sind. Über die erfolgte Planung ist zudem ein Vertrag zu schließen, welcher im Ernstfall auch als Nachweis der Rechtmäßigkeit des Penetrationstests dienen kann. Nach erfolgreicher Ausgestaltung des Umfangs und Vorgehens für den Test ist zunächst (privilegierter) Zugriff auf

das Testobjekt herzustellen, insofern dies nicht bereits vom Kunden bereitgestellt wurde (siehe Abschnitt 2.3.2, Seite 20). Hierfür können Login-Informationen von einem Nutzer oder Service erlangt werden, indem die verschiedenen Mechanismen, welche *Azure* für die Authentifikation bietet, betrachtet sowie Orte, an denen für gewöhnlich derartige Daten gespeichert sind, identifiziert werden. Darüber hinaus präsentiert Burrough Wege, die MFA als zusätzliche Absicherung einer Anmeldung zu umgehen [3, S. 1, 9].

Weiterhin setzt sich Burroughs Vorgehensweise damit fort, nach erfolgreich hergestelltem Zugriff wichtige und interessante Informationen innerhalb eines *Azure*-Abonnements zu ermitteln. Hierbei wird vorgeschlagen, zuerst etwaige Web-Services zu enumerieren. Anschließend sollten VMs, Speicherkonten und deren Zugangsschlüssel, öffentlich erreichbare Ports und Firewalls und zuletzt SQL-Server und -Datenbanken ermittelt werden. Dieser Schritt identifiziert jegliche Ressourcen, welche sich innerhalb eines Abonnements im Einsatz befinden, inklusive potenzieller Testsysteme, welche unter Umständen schwächere Sicherheitseinstellungen vorweisen. Entgegen dieser Reihenfolge wird allerdings mit dem Testen der Speicherkonten fortgefahren, da diese von einer Reihe von anderen Services für die Speicherung jeglicher Daten verwendet werden, beispielsweise virtuelle Datenträger von VMs oder Log-Dateien. Zudem werden diese Konten für das Teilen von Dokumenten und zur Speicherung von Backups verwendet, entsprechend einem *On Premise*-Dateiserver. Neben der zentralen Speicherung von Daten können auf diesem Weg die zwei Zugriffsschlüssel identifiziert werden, welche jedes derartige Konto aufweist. Diese liegen in vielen Fällen unverändert vor und sind somit potenziell auch in Quellcodes oder Konfigurationsdateien von Entwicklern hinterlegt [3, S. 35-36, 69-70].

Neben ihrem Eigenwert stellen Speicherkonten in *Azure* einen potenziellen Zugang zu VMs dar. Im weiteren Verlauf von Burroughs Vorgehensweise wird erklärt, wie auf die in Speicherkonten hinterlegten *Virtual Hard Disks* (VHD) einer oder mehrerer VMs zugegriffen werden kann beziehungsweise sensible und wichtige Daten aus diesen extrahiert werden können. Hierbei ist kein Zugriff auf das *Azure Portal* erforderlich. Aufbauend darauf besteht die Möglichkeit, die Passwortrücksetzungsfunktion der betroffenen VM auszunutzen, um vollen Zugriff auf diese und jegliche darauf ausgeführten Services zu erhalten sowie zugreifende Nutzer zu überwachen beziehungsweise Daten über diese zu sammeln [3, S. 91].

Im Anschluss fährt diese Vorgehensweise mit dem Testen von Netzwerken fort. Netzwerkkonnektivität ist ein zentraler Baustein in der Funktionsweise einer Cloud. In ihre Grundkonfiguration sind alle *Azure*-Services öffentlich erreichbar. Zudem bietet *Azure* die Möglichkeit, Verbindungen zwischen dem internen Netzwerk einer Organisation und eingesetzten Services herzustellen. Geschehen hier Fehlkonfigurationen, beispielsweise in den Firewalls, können sich diese schnell zu einem organisationsweiten Sicherheitsvorfall hochschaukeln. Dies hat den Grund, dass diese beiden Arten von Verbindungen, die öffentliche und interne Erreichbarkeit, zentral für die Realisierung

der Bedürfnisse des Kunden und dessen *Workloads* sind. Nachdem die essenziellen Services, welche die meisten Organisationen verwenden, untersucht wurden, widmet Burrough sich weiteren *Azure*-Services, angefangen bei *Azure KeyVault* als Speicher für Zertifikate und Schlüssel. Zudem werden Web-Applikationen, sprich *Azure App Service* und *Azure Automation* betrachtet. Mithilfe letzterem können Managementaufgaben im internen Netzwerk einer Organisation sowie in der Cloud automatisiert werden [3, S. 115-116, 139-140].

Zuletzt beleuchtet *Pentesting Azure Applications* Überwachungswerkzeuge, Logs sowie Benachrichtigungen und Alarme in *Azure*, mithilfe derer das für die Sicherheit zuständige Personal potenzielle Angriffe, insbesondere die in dieser Vorgehensweise beschriebenen, erkennen kann. Hier werden das ASC, die *Operations Management Suite* (OMS), das *Secure DevOps Kit* sowie weitere Methoden, Logs in *Azure* zu sammeln, erklärt [3, S. 163-164].

Cloud Penetration Testing Playbook

Diese Vorgehensweise wird von der CSA veröffentlicht und stellt eine Art Blaupause dar, nach der ein Penetrationstest innerhalb einer Cloud gegen die darin eingesetzten Services aufgebaut werden kann. Im Detail soll das gezeigte Vorgehen als Ausgangspunkt für das Testen derzeitiger Cloud-Technologien dienen, ausgehend von herkömmlichen Testmethodologien, um diese für die Cloud angemessen zu erweitern. Das Ergebnis stellt eine Art Checkliste dar, anhand derer wichtige Planungselemente, organisatorische Punkte sowie Testfälle Schritt für Schritt erledigt werden können [7].

Allgemein bietet die Vorgehensweise der CSA einen Überblick darüber, wie die Implementierungen von Services beziehungsweise Applikationen in der Cloud zu testen sind. Um die Anwendungen selbst zu testen, müsste nach wie vor zum Beispiel der *OWASP Testing Guide* (*Open Web Application Security Project*) herangezogen werden. Die CSA bietet somit keine technische Vorgehensweise, sondern eine schrittweise Anleitung, nach der ein Penetrationstest einer Cloud-Umgebung strukturiert werden kann. Vielmehr bietet die Richtlinie Hintergrundinformationen zu verschiedenen organisatorischen Aspekten um das Testen herum. Hierbei gliedert sich dieser Ansatz für Cloud-Penetrationstests in folgende Punkte (siehe Abbildung 4.10) [7]:

1: Testrahmen (orig. *Cloud Penetration Testing Scope*)

2: Testkontext (orig. *Cloud Penetration Testing in Context*)

3: Testziele (orig. *Cloud Penetration Testing Objectives*)

4: Testfälle und Bedenken (orig. *Cloud Penetration Test Cases and Concerns*)

- Vorbereitung (orig. *Preparation*)

- Bedrohungsmodellierung (orig. *Threat Modeling*)

- Aufklärung und Recherche (orig. *Reconnaisance and Research*)

- Testen (orig. *Testing*)

- Bericht (orig. *Report*)

5: Rechtliches (orig. *Legal*)

6: Training und Ressourcen (orig. *Training and Resources*)

7: Fazit (orig. *Conclusions*)

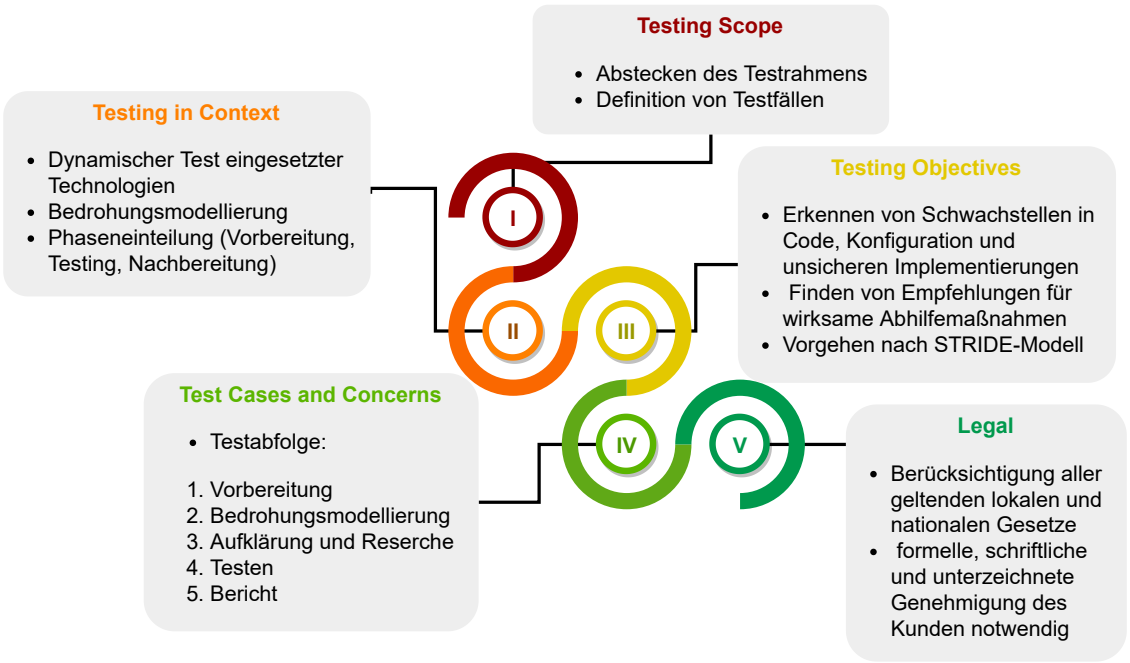

Abbildung 4.10: Ablauf und Inhaltspunkte des Cloud Penetration Testing Playbook [eigene Abbildung nach [7]]

Die CSA geht zu Beginn detailliert auf zahlreiche organisatorische Dimensionen, welche bei einem Cloud-Penetrationstest anfallen, ein. Hierbei wird umfassend der sich ändernde Testrahmen in Abhängigkeit von den jeweiligen Service-Modellen der Cloud dargestellt. Trotz der Inklusion von Applikationstests bei IaaS und PaaS limitiert die CSA den Rahmen eines Penetrationstests in ihrer Vorgehensweise jedoch auf drei grundsätzliche Bereiche: Konten- und Servicesicherheit sowie Businesslogik. Zusätzlich findet eine Einordnung eines Penetrationstests in den Kontext des *Microsoft Secure Development Lifecycle* statt. Die Verortung platziert den Penetrationstest im Abschnitt **Verifikation** (siehe Abbildung 4.10).

Zudem beschreibt die Vorgehensweise als Zielsetzung die Identifikation möglicher Sicherheitsrisiken und Schwachstellen in Cloud-Umgebungen, geordnet nach dem STRIDE-Modell von *Microsoft*, weshalb auch die beschriebenen Testfälle nach diesem Muster angelegt sind: ***S**poofing*, ***T**ampering*, ***R**epudiation*, ***I**nformation Disclosure*, ***D**enial of service*, ***E**levation of Privileges* [7].

Die **Vorbereitungsphase** widmet sich der Ausarbeitung notwendiger Vereinbarungen mit dem Kunden des Tests sowie der Festlegung des Testrahmens. Bei Bedarf, jedoch nicht im Fall von *Azure*, wäre hier eine Einverständniserklärung des *Providers* einzuholen. Anschließend sind in dieser Phase TTPs auszuwählen, sowohl für die Cloud betreffende als auch andere Testfälle. In der anschließenden **Bedrohungsmodellierungsphase** liegt das Hauptaugenmerk darauf, die Wünsche des Kunden in Bedrohungsmodelle umzuwandeln und somit auf den Testrahmen und *Provider* zugeschnittene Bedrohungen zu ermitteln [7].

In der Phase **Aufklärung und Recherche** sind vielfältige Taktiken für den Informationsgewinn einzusetzen. Dies beinhaltet zum Beispiel die Abfrage von DNS-Einträgen, das Durchsuchen von *Code Repositories* nach Zugangsdaten, die Enumeration von Nutzern und insbesondere Administratoren inklusive Logindaten, Speicherkonten sowie die allgemeine Festlegung von hochwertigen Zielen und die Recherche nach Schwachstellen oder Fehlkonfigurationen. Zuletzt sollen diese Informationen mit den zuvor ermittelten Bedrohungsmodellen in Verbindung gebracht werden. Daraufhin hat die **Testphase** das Ziel, nach dem STRIDE-Modell verschiedene potenzielle Schwachstellen zu verifizieren, indem beispielsweise versucht wird, verschiedene *Spoofing*-Szenarien zu simulieren (*Man-in-the-Middle*, *Session Hijacking*, *Domain Hijacking*, ...), Datensätze oder Services zu verändern (*Tampering*) und nach diesem Muster die einzelnen Aspekte von STRIDE zu verifizieren [7].

Abgeschlossen wird die Vorgehensweise der CSA mit einem **Bericht**. Hierbei sollen die zentralen Erkenntnisse des Penetrationstests rekapituliert werden und gegen die gültigen Industriestandards und *Best Practices* bewertet werden. Zudem ist ebenfalls die Durchführung des Tests zu bewerten, beispielsweise hinsichtlich der Abdeckung des Testrahmens oder der Testziele [7]. Der Grundaufbau der CSA für einen Penetrationstest in der Cloud ähnelt der Vorgehensweise des BSI für einen traditionellen Penetrationstest enorm. Dies stellt keine große Umstellung für einen Tester dar, lediglich die einzigartigen Gegebenheiten, welche eine spezifische Cloud mit sich bringen würde, sind in diesem Fall als neu anzusehen.

4.3.2 Einsatzempfehlung

Ausgehend von den präsentierten Vorgehensweisen für Penetrationstests in *Azure* beziehungsweise der Cloud können eine Reihe von Richtlinien abgeleitet werden, welche die Empfehlung der einzelnen Methodiken begründen. Diese wird in Tabelle 4.7 nochmals entsprechend verdeutlicht. Hierbei handelt es sich um **Technische Tiefe**, **Organisatorische Tiefe**, **Serviceorientierung**, **Schwachstellenorientierung** sowie **Provider** und **Testumfang**.

Burroughs Vorgehensweise in *Pentesting Azure Applications* illustriert im Detail die Durchführung der einzelnen Schritte eines Penetrationstests bei einem *Azure*-Mandanten, inklusive der Ein- und Ausgabe bei verwendeten Tools. Darüber hinaus werden umfangreiche Begründungen für die einzelnen Testfälle dargestellt. Deshalb weist dieses Vorgehen eine hohe technische Tiefe auf, inklusive umfassender Hintergrundinformationen sowohl für den Tester als auch für die Behebung einzelner potenzieller Sicherheitslücken in Form von *Best Practices*. Auch die organisatorische Seite eines Penetrationstests in *Azure* wird beleuchtet, indem das *Scoping*, also die Festlegung des Testumfangs, in den Vordergrund gerückt und betont wird. Ebenso folgt Burrough einer klaren Struktur, orientiert an den meistgenutzten (wichtigsten) Services, welche nahezu immer anzutreffen sind. Daher verfügt diese Vorgehensweise auch über eine praxisnahe organisatorische Tiefe. Ausgerichtet ist *Pentesting Azure Applications* strikt auf *Azure*, wodurch eine detailliertere Beschreibung und Analyse der einzelnen Services und Testtechniken vorgenommen werden kann.

Zu empfehlen ist *Pentesting Azure Applications*, wenn ein service-/ressourcenorientierter Penetrationstest bei einem *Azure*-Mandanten beziehungsweise dessen Abonnements durchgeführt werden soll. Hierbei liegt eine klare, strukturierte Durchführungsanweisung vor, welche bedarfsgerecht auf die zu testenden Services des Kunden angepasst werden kann. Ebenso bietet Burrough eine Hilfestellung, welche Tools für die einzelnen Testabschnitte und Fälle genutzt werden können. Insbesondere kann Burrough für Penetrationstests von *Azure*-Mandanten an sich sowie für einen speziellen IaaS-Test verwendet werden.

Die CSA bietet mit dem *Cloud Penetration Testing Playbook* geringe technische Tiefe, da die Hintergründe und Durchführungsbeschreibungen der aufgegriffenen Testfälle nicht im Detail berücksichtigt werden. Umgekehrt weist diese Vorgehensweise eine sehr hohe organisatorische Tiefe auf, da die Einteilung eines Cloud-Penetrationstests, die potenziellen Testfälle, Fokuspunkte und Sicherheitsdimensionen genauer erläutert und dargestellt werden. Darüber hinaus ordnet die CSA ihre Methodik nicht in Cloud-Services, sondern in einen phasenweisen Aufbau ähnlich dem des BSI und beschreibt gleichzeitig eine schwachstellenorientierte Herangehensweise nach dem STRIDE-Modell.

Zudem richtet sich das Dokument nicht spezifisch an *Azure*, sondern kann universell bei den Cloud-Infrastrukturen aller *Provider* eingesetzt werden. Aus diesem Grund beschränkt sich der Testumfang daher nicht auf spezifische (wichtige) Services, sondern versucht, eine möglichst komplette Abdeckung aller potenziellen Testfälle zu erzielen.

Zu empfehlen ist das *Cloud Penetration Testing Playbook*, wenn eine umfassende organisatorische Grundlage für einen *Azure*-Penetrationstest genutzt werden soll, welche möglichst viele Testfälle abdeckt und das Potenzial aufweist, alle Arten von denkbaren Schwachstellen zu erfassen. Auf dieser Basis besteht hier die Möglichkeit, andere *Testing Guides* für die Durchführung einzelner Testfälle bei verschiedenen Services, beispielsweise Web-Anwendungen, heranzuziehen. Diese Vorgehensweise eignet sich im Grunde universell für jedes Service-Modell und für das Testen des Mandanten, je nach Einsatz müssen die Testfälle jedoch bedarfsgerecht ausgewählt und weitere Frameworks hinzugezogen werden.

Tabelle 4.7: Bewertung und Empfehlung der Hardening-Vorgehensweisen nach zentralen Kriterien [eigene Tabelle]

		Vorgehensweisen	
		Pentesting Azure Applications	CSA Playbook
Kriterien	Technische Tiefe	✓	(✓)
	Organisatorische Tiefe	(✓)	✓
	Serviceorientierung	✓	
	Schwachstellenorientierung		✓
	Provider	Azure	Universell
	Testumfang	Wichtigste Services	Komplett

✓ Erfüllt das Kriterium
(✓) Erfüllt das Kriterium teilweise

4.3.3 Zu nutzende Tools

Wie für den *Hardening Check* existieren auch für einen Penetrationstest in *Azure* Werkzeuge, welche bei der Durchführung eines Tests verwendet werden können, um die einzelnen Testfälle zu bearbeiten und mit dem *Azure*-Mandanten, dessen Abonnements und Services in gegenseitigem Austausch zu stehen [3, S. xxiv].

Windows-Betriebssystem

Da es sich bei *Azure* um ein Produkt der Firma *Microsoft* handelt, ist eine Vielzahl der potenziell zu verwendenden Tools auf ein Windows-Betriebssystem angewiesen. Hierbei ist *Windows 7* die mindestens erforderliche Version, jedoch ist die Mehrzahl der

Werkzeuge basierend auf demselben Prozess, mit dem *Azure* aktualisiert wird, ebenfalls erweitert und angepasst worden, sodass eine Funktion auf höheren Versionen von Windows anzunehmen ist [3, S. xxiv]. An den nachfolgenden Beispielen *PowerShell-Module/-Skripte* und *Azure Storage Explorer* wurde die Funktion unter *Windows 10*, der derzeit aktuellsten Version des Windows-Betriebssystems, nachgewiesen.

PowerShell-Module/-Skripte

Für die Aufgabe der Informationsgewinnung und Aufklärung bei einem *Azure*-Mandanten beziehungsweise einzelner Abonnements wird die Nutzung der *PowerShell* und des zugehörigen Moduls *Azure PowerShell* empfohlen. *Azure PowerShell* erlaubt die Authentifikation auf allen bekannten Wegen bei einem *Azure*-Mandanten sowie die anschließende Enumeration der enthaltenen Abonnements. Hierbei können Abonnements betrachtet, enthaltene Ressourcengruppen und Ressourcen aufgelistet sowie gemäß ihres Typs, beispielsweise *App Services* (Web-Anwen-dungen) betrachtet und auf ihre Instanzen, Standard-Dokumente etc. überprüft werden [3, S. 40-50]. Auch Sicherheitsmaßnahmen, beispielsweise SQL-Firewall-Regeln oder Datenbanken eines SQL-Servers können auf diesem Weg aufgedeckt werden [3, S. 61-62].

Azure Portal

Das *Azure Portal* als grafische Verwaltungsoberfläche eines *Azure*-Mandanten bietet ebenfalls eine Möglichkeit, die mittels *Azure PowerShell* durchführbaren Aufgaben zu bearbeiten. Mit validen Logindaten, beispielsweise von einem *Azure AD*-Konto, besteht hier die Möglichkeit zur Authentifizierung und der anschließenden Sammlung von Informationen über Abonnements, Ressourcen, laufende Services etc. Zudem können die Rechte eines Nutzers ausgeschöpft und beispielsweise das Erzeugen oder Modifizieren von Ressourcen getestet werden.

Azure Storage Explorer

Mithilfe dieses Tools kann lokaler Zugriff auf etwaige Speicherkonten in *Azure* hergestellt werden. Hierbei bestehen verschiedene Möglichkeiten der Authentifizierung, beispielsweise über Logindaten, vorausgesetzt der Nutzer ist privilegiert genug, oder über so genannte *Storage Account Keys*. Diese können in anderen Services, im Quellcode von Entwicklern oder auch im *Storage Explorer* eines Entwicklers gefunden werden [3, S. 74, 75]. Allerdings gelten Logindaten eines Nutzers als die nutzbringendere Methode [3, S. 84]. Mithilfe des Zugriffs auf die Speicherkonten eines *Azure*-Mandanten beziehungsweise Abonnements können unter Umständen weitere Nutzerdaten, *Secrets*, Zugriffsschlüssel oder VM-Festplatten gefunden werden [3, S. 84-85, S.92-93]. Folgende Abbildung 4.11 zeigt eine Ansicht des *Azure Storage Explorers*.

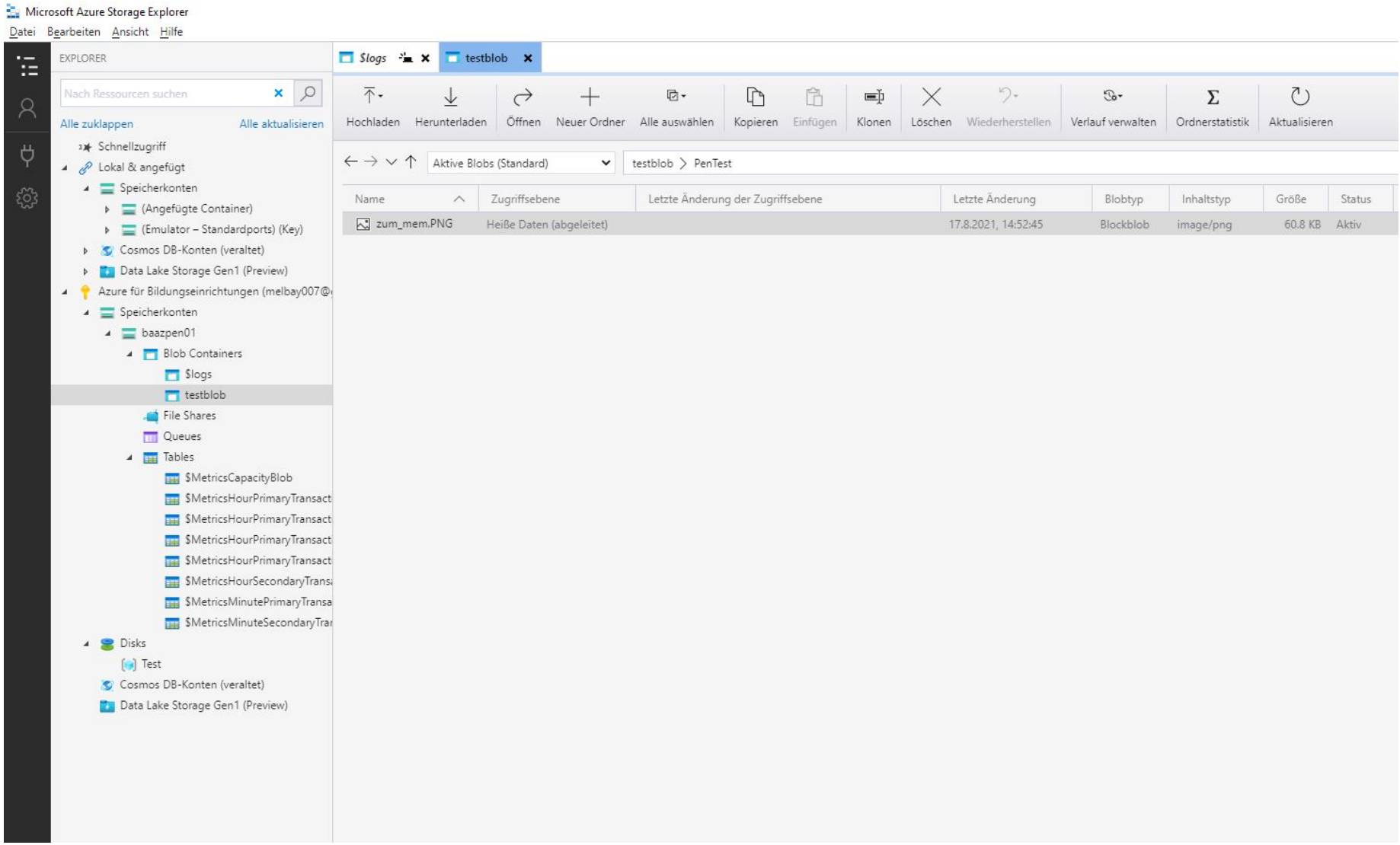

Abbildung 4.11: Ansicht eines Storage Blobs mit untergeordnetem Ordner und Bild als Inhalt im Azure Storage Explorer [Screenshot]

Autopsy

Besteht Zugriff auf ein Speicherkonto, welches eine oder mehrere VHDs beherbergt und besteht die Möglichkeit, diese herunterzuladen, können diese analysiert werden, um nützliche Informationen zu gewinnen. Hierbei kann es sich einerseits um *Findings* des Penetrationstests selbst handeln, beispielsweise sensible Daten, oder aber um Passwörter und andere Möglichkeiten, Zugriff auf die zugrundeliegende VM zu erhalten. Die Analyse einer VHD kann potenziell mit *Autopsy* durchgeführt werden [3, S. 94-95].

Weitere Tools

Neben den zuvor dargestellten Werkzeugen existieren unzählige weitere, welche während eines Penetrationstests in *Azure* verwendet werden können. Beispielsweise können VHDs unter Windows (und Linux) auf normalem Weg zur Analyse als virtuelle Datenträger eingebunden werden. Die Sicherung von Logindaten auf Windows-Rechnern im Testrahmen kann mittels *mimikatz* durchgeführt werden, während Hashes mit der Hilfe von *hashcat* gebrochen werden können [3, S. 95, S. 100-101].

4.4 Anwendbarkeit von Sicherheitstests bei den verschiedenen Service-Modellen

Zwar können beide Arten von Sicherheitstests, *Hardening Checks* und Penetrationstests, in *Azure* durchgeführt werden, jedoch müssen hierbei Differenzen bezüglich der Anwendbarkeit in verschiedenen Situationen berücksichtigt werden. Insbesondere die einzelnen Service-Modelle einer Cloud müssen hierbei evaluiert und als Faktoren in Betracht gezogen werden. Dies hat den Grund, dass sich in Abhängigkeit dieser Modelle unterschiedliche Verantwortungsbereiche die IT-Sicherheit betreffend für den Kunden und *Cloud Provider* herauskristallisieren (siehe Abbildung 2.1, Seite 11).

4.4.1 Allgemeine Empfehlungen

Ein *Black Box*-Test sollte von vornherein stets ausgeschlossen werden. Dies hat den Grund, dass ein solcher Test mit wenig bis keinen Informationen startet, abgesehen von dem Namen des Kunden und anderen Anhaltspunkten für die Identifikation weiterführender Informationen über das Ziel. Aufgrund der geteilten Natur jeglicher physischer sowie vereinzelter logischer Ressourcen, beispielsweise IP-Adressen, ist das Auseinanderhalten zweier verschiedener Mandanten in *Azure* nicht immer einfach. Sollten irrtümlich andere *Azure*-Ressourcen als die des eigentlichen Kunden überprüft oder angegriffen werden, stellt dies einerseits einen klaren Verstoß gegen rechtliche sowie

einen Bruch verschiedener technischer und fachlicher Rahmenbedingungen dar. Dies gilt ebenfalls für einen versehentlichen Test der (physischen) Komponenten unter der Kontrolle des *Cloud Providers*.

Während ein Penetrationstest durchaus auf einzelne Services oder Ressourcen ausgerichtet werden kann, ist dies kein empfehlenswertes Vorgehen. Eine einzelne VM beispielsweise bietet für sich genommen eine geringe Angriffsfläche und ein Test würde vergleichsweise wenig Ergebnisse liefern, um die Sicherheit dieser zu verbessern. Dies ist beispielsweise eine Folge davon, dass der Einsatz einer VM in *Azure* so konfiguriert wird, dass von Beginn an lediglich ein einziger Zugang zu dieser besteht, beispielsweise über SSH (*Secure Shell*) oder RDP (*Remote Desktop Protocol*). Insofern dieser Service korrekt konfiguriert wurde und aktuell gehalten wird, geht hiervon nahezu keine Gefahr aus. Die Bedrohung läge jedoch zum Beispiel darin, dass die Zugangsdaten eines Nutzers unzureichend sicher gewählt sind, oder bei anderen Services wiederverwendet werden. Eine Ausdehnung des Testrahmens auf das *Azure AD* oder das *Azure*-Konto eines Kunden würde diesen Umstand berücksichtigen und möglicherweise aufdecken.

Eine ähnliche Empfehlung ist für einen *Hardening Check* auszusprechen. Prinzipiell sollte ein solcher Sicherheitstest stets ausgedehnt auf den gesamten *Azure*-Mandanten durchgeführt werden. Dies hat den Grund, dass die Überprüfung und Sicherung einzelner Services zwar möglich ist, die verschiedenen Dimensionen der Sicherheit in *Azure* jedoch miteinander interagieren und eng verbunden sind. Beispielsweise kann für SaaS-Anwendungen die Anmeldung über ein *Azure AD*-Konto genutzt werden, womit die Sicherheit der Applikation an die der *Azure AD*-Konfiguration gebunden wird.

Für beide Arten von Sicherheitstests gilt, dass eine regelmäßige Durchführung und Wiederholung sehr zu empfehlen ist. Auf diese Weise wird das bestehende Sicherheitsniveau beständig evaluiert und immer neu bewertet, um aus den erfassten Ergebnissen neue Schlussfolgerungen und Verbesserungen abzuleiten.

4.4.2 Bei SaaS

Die Verantwortung des Kunden für die Sicherheit bei SaaS sowie der Bereich dessen, was dieser kontrollieren kann, fallen sehr gering aus (siehe Abbildung 2.1, Seite 11). Kontrolliert wird lediglich der Aspekt *Daten und Zugriff*, was eine Sicherheitsverantwortung für Daten-Governance, Rechtemanagement, Client-Endpunkte, sowie Konten und deren Zugriffsregeln und -rechte bedeutet. Daraus resultierend ergibt sich ein übersichtlicher Testrahmen beziehungsweise -umfang.

Bei jeglichen Anwendungen, welche in den Definitionsbereich SaaS fallen, liegt die Hauptverantwortung bei demjenigen, der diese Anwendungen bereitstellt, in diesem Fall *Microsoft* oder ein so genannter Dienstanbieter, welcher seine Software zur Nutzung in *Azure* anbietet. Da der Kunde und Nutzer derartiger Anwendungen lediglich Kontrolle über darin verarbeitete Daten sowie erstellte oder authentifizierte Nutzer der Anwendung hat, bietet sich für einen Penetrationstest generell keine externe Testgrundlage. Dies hat den Grund, dass jegliche traditionelle Testfälle, beispielsweise *SQL Injections* bei einer Datenbank oder *Brute Forcing* bei einer Anmeldung, die zugrundeliegende Infrastruktur der Applikation betreffen und potenziell beeinträchtigen können. Dies stellt eine Verletzung des Testrahmens und der *Rules of Engagement* dar und übertritt somit die geltenden Rahmenbedingungen.

Ein *White Box*-Test könnte als interne Überprüfung der von dem Kunden kontrollierten Parameter durchgeführt werden, beispielsweise ein Test der bestehenden Nutzerberechtigungen, um potenziell zu schwache, weitgefasste oder nicht erforderliche Berechtigungen nachzuweisen, welche unerlaubte Aktivitäten zulassen und diese mit einem PoC zu belegen. Dies stellt jedoch die einzige Art Penetrationstest dar, welche bei SaaS praktikabel wäre. Der Wert der Ergebnisse ist nicht von der Hand zu weisen, da Sicherheitsprobleme in Zusammenhang mit Berechtigungen und potenziell unzureichend abgesicherten Abläufen innerhalb einer SaaS-Anwendung aufgedeckt werden könnten. Für einen *Hardening Check* bietet sich bei SaaS kaum eine Grundlage zur Durchführung, da der Kunde geringen Einfluss auf sicherheitsrelevante Konfigurationseinstellungen der Anwendung ausüben kann. Dies bedeutet, dass der zeitliche Aufwand zwar gering wäre, die erzielten Ergebnisse jedoch kaum sinnvolle Einblicke und Anregungen für die Verbesserung der Applikationssicherheit bieten würden. Der *White Box*-Test kann zudem von dem Tester allein durchgeführt werden, während für einen *Hardening Check* mit einem Verantwortlichen des Kunden zusammengearbeitet werden müsste. Somit ist der Aufwand für den Kunden in diesem Fall höher.

4.4.3 Bei PaaS

Im Fall von PaaS dehnt sich der Kontrollbereich des Kunden über die Aspekte *Daten und Zugriff* sowie *Anwendungen* aus. Auf die Sicherheitsverantwortung übertragen bedeutet dies das Aufkommen für Daten-Governance, Rechtemanagement, Client-Endpunkte, Konten- und Zugriffsrechte sowie teilweise Identitäts- und Verzeichnisinfrastruktur, Anwendungen und Netzwerkkontrollen (siehe Abbildung 2.1, Seite 11). Letztere drei Bereiche werden mit dem *Cloud Provider* geteilt. Der Testrahmen ist moderat gesteigert, da hier Teile der Anwendungen und des Netzwerks einer gemieteten Plattform in die Mitverwaltung des Kunden gegeben werden.

Der *Azure App Service* für das schnelle Deployment von Web-Applikationen ist ein Beispiel für eine PaaS-Anwendung in *Azure*. Ein Penetrationstest einer solchen Web-Applikation könnte mit einem traditionellen Web-Penetrationstest gleichgesetzt werden. Dies legt die Durchführung eines *Grey Box*-Tests nahe, um sowohl von innen als auch von außen etwaige Testfälle durchzuführen. Aufgrund dessen, dass *Microsoft* lediglich eine Plattform für die Entwicklung und den Einsatz von Webanwendungen und anderen Applikationen bereitstellt, ist eine Einbeziehung jeglicher Komponenten innerhalb einer Anwendung durchführbar und zu empfehlen. Die Grenzen eines Penetrationstests liegen bei Überprüfungen der zugrundeliegenden Elemente, beispielsweise dem Betriebssystem eines Hosting-Servers für eine Webanwendung oder Datenbank. Beispiele für mögliche Testfälle wären eine Evaluation der Sicherheit von Login-Methoden, der Verschlüsselung von Datenbanken beziehungsweise deren Inhalt oder die Berechtigungen einzelner Nutzer innerhalb der Anwendung.

Auch ein *White Box*-Test könnte bei PaaS durchgeführt werden. Das Ziel läge hierbei ähnlich wie bei SaaS beispielsweise auf einer internen Verifikation von Nutzerberechtigungen, aber auch in einer Betrachtung des Quellcodes Applikation, insofern dieser durch den Kunden selbst erstellt wurde. Die Geschäftslogik der Anwendung kann in diesem Zusammenhang ebenfalls einer Überprüfung unterzogen werden. Der Aufwand ist logischerweise höher anzusetzen, als der eines Grey Box-Tests, jedoch bieten beide wertvolle Ergebnisse und einen Überblick über potenzielle Sicherheitslücken und Risiken bei mittels PaaS entwickelten und eingesetzten Applikationen. Der *White Box*-Test würde allerdings nur eine Innenperspektive bieten. Sollte das entsprechende Ziel öffentlich erreichbar sein, also Schnittstellen zum Internet hin offenbaren, ist ein *Grey Box*-Test als empfehlenswerter anzusehen.

Für einen *Hardening Check* bietet sich hier ähnlich wie bei SaaS keine ausgedehnte Grundlage. Die Applikation, welche mittels PaaS bereitgestellt wird, kann natürlich einer Überprüfung hinsichtlich der in *Azure* zu treffenden Konfigurationseinstellungen unterzogen werden, beispielsweise für den Nutzerzugriff auf diese. Der Aufwand ist aufgrund des nach wie vor moderaten Testrahmens nicht hoch und etwaige Ergebnisse dienen stets der Verbesserung des Sicherheitsniveaus.

4.4.4 Bei IaaS

Zuletzt erstreckt sich bei IaaS der Testrahmen über jegliche Bereiche, welche nicht Teil der physischen Infrastruktur von *Azure* sind. Dies bedeutet eine Betrachtung der Bereiche *Daten und Zugriff*, *Anwendungen*, *Runtime*, *Betriebssystem* und *Virtuelle Computer/Hosts*. In diesem Verantwortungsbereich befinden sich somit Daten-Governance, Rechtemanagement, Client-Endpunkte, Account- und Zugriffsmanagement, Identitäts- und Verzeichnisinfrastruktur, Anwendungen, Netzwerkkontrollen sowie etwaige Betriebssysteme (siehe Abbildung 2.1, Seite 11). Dies ist von allen Service-Modellen der am Weitesten gefasste Testrahmen und umschließt jegliche virtuelle Komponenten, die ein *Azure*-Mandant beinhalten kann.

Ein IaaS-Test hat eine große Ähnlichkeit mit einem gewöhnlichen Infrastruktur-Penetrationstest. Der einzige Unterschied liegt in der Virtualisierung der Teilnehmer und Komponenten der Infrastruktur. Ein *Grey Box*-Test bietet hier aller Wahrscheinlichkeit nach den besten Ansatz, um einen möglichst vielseitigen Eindruck von allen Aspekten bei IaaS zu erhalten. Auf diesem Weg ergibt sich eine Überprüfung der inneren Bestandteile wie beispielsweise VMs, Datenbanken, Web-Servern und dergleichen, während öffentliche Zugangspunkte beziehungsweise Endpunkte auf potenzielle Angriffswege von außen verifiziert werden können. Ein *White Box*-Test ist anwendbar, jedoch sollte hier der variable Aufwand je nach Größe der Infrastruktur beziehungsweise des gewünschten Testrahmens berücksichtigt werden. Jegliche Tests, welche Einfluss auf das physische Netzwerk oder den *Hypervisor* und jegliche darunterliegenden Komponenten nehmen könnten, sind zu unterlassen.

Für einen *Hardening Check* eröffnet sich bei IaaS erstmals ein geeigneter Testrahmen und Einsatzbereich, da hier jegliche virtuellen Aspekte der Infrastruktur dem Kunden unterstellt sind und sich hieraus die größte Reichweite von Einstellungen und Konfigurationsmöglichkeiten ergibt, welche bedacht werden muss, um die einzelnen Wechselwirkungen zwischen Komponenten der Infrastruktur zu berücksichtigen. Beispielsweise besteht ein hoher Bedarf an einem ordnungsgemäß konfigurierten Identitätsmanagement sowie an sicheren virtuellen Netzwerken. Der *Hardening Check* ist sicherlich mit dem höchsten Aufwand verbunden, auch wenn einzelne Überprüfungen automatisiert durchgeführt werden könnten, da die Menge an zu berücksichtigender Dokumentation und Besprechung mit einem Verantwortlichen des Kunden den Zeitaufwand steigert. Ähnlich hoch ist der Aufwand eines *White Box*-Tests, allerdings ist hierbei vonseiten des Kunden nur sämtliches zu nutzendes Dokumentations- und Spezifikationsmaterial zur Verfügung zu stellen. Auch der *Grey Box*-Test könnte durch einen großen Testrahmen und eine hohe Zahl an Testfällen in dessen Aufwand steigen, jedoch bietet diese Variante gleichzeitig einen möglichst umfassenden Rundumblick über die Sicherheit bei IaaS. Würde ein *Hardening Check* zuvor durchgeführt, ließe sich direkt die Effektivität der getroffenen Sicherheitsvorkehrungen und Konfigurationseinstellungen mit bewerten.

4.4.5 Spezialfall I: On Premises

Dieses Deployment-Modell ist deshalb ein Spezialfall, weil der Kunde hierbei jegliche Hardware selbst bereitstellt und somit ein eigenes Rechenzentrum betreibt. Hierfür wird der *Azure Stack* außerhalb der *Public Cloud* in einem lokalen Rechenzentrum betrieben, um *Azure*-Dienstleistungen nutzen zu können, welche problematisch in der *Public Cloud* zu betreiben wären, beispielsweise aufgrund von gesetzlichen Vorgaben oder technischen Voraussetzungen [21]. Jegliche Sicherheitsverantwortung, inklusive der physischen Netwerkinfrastruktur, Server und des Rechenzentrums, liegen somit in der Hand des Kunden. Diese einzigartige Position erhebt keine Limitierungen bei der Auswahl und Durchführung des Testverfahrens. Hierbei ist deshalb auch ein *Black Box*-Test, potenziell inklusive *Social Engineering*, denkbar. Der Testrahmen umfasst alle Schichten der virtuellen und physischen Komponenten der *Azure*-Infrastruktur (siehe Abbildung 2.1, Seite 11).

Für die Testarten des Penetrationstests bestehen in diesem Szenario keine Limitationen, auch *Grey Box*- und *White Box*-Tests können durchgeführt werden. Letzterer erfordert den potenziell höchsten Aufwand der drei Testarten, da die Menge der zur Verfügung stehenden Hilfsmittel wie beispielsweise Dokumentationen sehr umfassend ist. Der zweithöchste Aufwand ist bei einem *Black Box*-Test anzunehmen, da hierbei keine weiteren Informationen, abgesehen von einem Startpunkt für den Test, geboten werden und jegliche weitere Faktoren durch den Tester ermittelt werden müssen. Der *Grey Box*-Test ist auch hier die Variante, welche gemessen an dem erforderlichen Aufwand und den potenziell erzielbaren Ergebnissen die beste Kosten-Nutzen-Relation aufweist. Zu berücksichtigen ist hierbei, dass die Sicherheit des Rechenzentrums nicht länger in der Verantwortung von *Microsoft* liegt. Eine Betrachtung dieser physischen Aspekte ist also ebenfalls als lohnenswert anzusehen, um ein möglichst hohes Sicherheitsniveau zu garantieren.

Auch für einen *Hardening Check* ergeben sich hierbei keine Limitationen. Jegliche Aspekte innerhalb der Verantwortlichkeit des Kunden können nach bekannten Vorgehensweisen betrachtet und überprüft werden. Bezüglich der Ergebnisse ist anzunehmen, dass ein der *Hardening Check* insbesondere hier, ohne Teilung der Verantwortung mit dem *Cloud Provider*, nützliche Ergebnisse liefert, um eine stabiles Ausgangssicherheitsniveau zu bieten, welches kontinuierlich durch Penetrationstests überprüft und anschließend verbessert werden kann.

4.4.6 Spezialfall II: Azure-Konto

Das *Azure*-Konto ist deshalb ein Spezialfall, da an diesem ein separater Penetrationstest durchgeführt werden kann. Hier ist das Ziel, die Verwundbarkeit des Kontos als Angriffsvektor auf etwaige Ressourcen innerhalb der einzelnen Abonnements nachzuweisen. Die Grundvorgehensweise hierbei sieht einen *Grey Box*-Test vor, bei dem der Zugang zu dem *Azure*-Konto oder einem Konto innerhalb des *Azure AD* des Mandanten gewährt wird. Ausgehend hiervon wird versucht, einzelne Ressourcen zu kompromittieren. Dieser Ansatz eröffnet einen weiteren Blickwinkel auf die Sicherheit in *Azure*, da hierbei nicht von einem Angriff auf einen Service oder eine Ressource, sondern von einem Angriff auf die gemeinsame Managementinstanz eines *Azure*-Mandanten ausgegangen wird. Der Aufwand eines solchen Penetrationstests misst sich daran, wie gut die einzelnen Ressourcen geschützt sind sowie wie der jeweilige Einzelschutz ineinander übergreift, um Synergieeffekte zu bilden. Dies kann sowohl im positiven als auch im negativen Sinne geschehen. Wird beispielsweise der Schlüssel eines *Azure*-Speicherkontos in dem Quellcode einer mittels PaaS eingesetzten Anwendung hart kodiert, ist dies ein Sicherheitsrisiko, welches sich auf alle anderen Services, welche dieses Speicherkonto nutzen, auswirkt.

Ein *Hardening Check* des *Azure*-Kontos als gesamte Einheit bietet den größten Testrahmen, jedoch auch den größten Effekt. Die in dieser Arbeit diskutierten *Hardening*-Vorgehensweisen sind darauf ausgerichtet, universell einsetzbar zu sein und keinen Fokus auf ein spezifisches Service-Modell zu legen, sondern darauf zu achten, dass ein Gesamt-Sicherheitsniveau erzielt wird. Dementsprechend findet der *Hardening Check* auf diese Weise die beste Anwendung und ist für diesen Einsatz sehr zu empfehlen.

5 Diskussion

In diesem Kapitel werden die zuvor in Kapitel 4 (siehe Seite 45) dargestellten Ergebnisse diskutiert und interpretiert. Hierfür werden zuerst die Vorgehensweisen für einen *Hardening Check* und anschließend die für einen Penetrationstest untereinander verglichen und bewertet. Anschließend wird zudem die Bewertung der Anwendbarkeit dieser zwei Arten von Sicherheitstests bei den drei unterschiedlichen Service-Modellen hinsichtlich ihrer Aussagekraft betrachtet. Zuletzt werden drei andere Quellen aus diesem Bereich mit dieser Arbeit bezüglich ihrer Gemeinsamkeiten und Unterschiede überprüft.

5.1 Interpretation der Ergebnisse

5.1.1 Hardening Check: ausgewählte Vorgehensweisen

Die Methode von Toroman und Janetscheck verfügt über umfangreiches, fundiertes Wissen, welches einzelne Maßnahmen zum *Hardening* in *Azure* mit Erklärungen und Hintergrundinformationen versieht. Mithilfe der einzelnen Kapitel besteht die Möglichkeit, die Sicherheit und Härtung eines *Azure*-Mandanten von Grund auf zu überprüfen, angefangen bei dessen organisatorischer Einteilung, der Anwendung von Sicherheitsrichtlinien bis hin zu jeglichen ressourcenbezogenen oder globalen Sicherheitsmaßnahmen. Hierbei werden Empfehlungen zu Identitätsmanagement, Netzwerksicherheit und Datensicherheit gegeben sowie in *Azure* integrierte Tools erklärt, welche bei der Evaluierung und Garantie der Sicherheitsposition hilfreich sind.

Nachteilig für Toromans und Janetschecks Buch gilt festzuhalten, dass die darin beschriebene Vorgehensweise kein *Benchmark* im bestehenden Sinne darstellt, sondern Empfehlungen, welche auf den Erfahrungswerten der Autoren basieren. Somit besteht hier aller Wahrscheinlichkeit nach keine akkreditierte Anerkennung beziehungsweise keine Akzeptanz als Industriestandard. Dennoch ist eine Kenntnis und Anwendung bei einem *Hardening Check* zu empfehlen, um das tiefere Verständnis hinter einzelnen Maßgaben und Hinweisen zu schärfen und somit das *Warum* zu überprüfender Punkte anzusprechen.

Der ASB von *Microsoft* bietet einen direkten Einstieg in das *Hardening* von *Azure* über integrierte Werkzeuge, wie beispielsweise das ASC oder *Azure Sentinel*. Die hierin enthaltenen Kontrollen und Empfehlungen zielen darauf ab, größtmöglichen positiven Einfluss auf die Sicherheit eines *Azure*-Mandanten auszuüben und von Anfang an ein hohes Sicherheitsniveau zu erzielen. Sämtliche Einträge im ASB sind mit Detailangaben versehen, warum die jeweilige Kontrolle durchzuführen wäre. Zudem sind jeder Emp-

fehlung entsprechende Gegenstücke aus den CIS-Kontrollen v7.1 sowie aus der NIST SP 800-53 beigefügt. Somit zeigt sich, dass der ASB anerkannten Industriestandards folgt.

Jedoch auch wenn die Kontrollen des ASB sich beispielsweise auf die des CIS abbilden lassen, bedeutet dies nicht, dass die *Hardening*-Empfehlungen des CIS in vollem Umfang umgesetzt werden. Somit wäre hier eine separate Überprüfung durchzuführen. Empfehlenswert wäre dementsprechend eine Verwendung des ASB in Kombination beispielsweise mit dem *Microsoft Azure Foundations Benchmark*.

Dieser bietet ebenfalls eine solide Grundlage, um die Sicherheit eines *Azure*-Mandanten zu überprüfen und zu verbessern, jedoch wird explizit darauf hingewiesen, dass dieses Sicherheitsniveau lediglich eine gesunde Ausgangslage bietet, auf welcher bedarfsorientiert weitere Maßnahmen ergriffen werden sollen. Zudem ist dieser *Benchmark* des CIS keine vollständige Abbildung sämtlicher möglicher Konfigurationsentscheidungen und -einstellungen. Nichtsdestotrotz wird auf diesem Weg eine Basis für ein sicheres System geschaffen.

Im Allgemeinen behandeln die drei Vorgehensweisen dasselbe Thema und verfügen somit über sich deckende Fokuspunkte und zentrale Positionen. Während allerdings Toroman und Janetscheck beispielsweise *Governance* und die damit verbundenen Planungen zu Strategie und Richtlinien als Basis für ein umfassendes *Hardening* ansehen, werden hierzu passende Empfehlungen im ASB als Letztes erwähnt. Hierbei ist festzustellen, dass der ASB die Aufmerksamkeit auf das Netzwerk als den Perimeter setzt, gefolgt von der sicheren und lückenlosen Verwaltung von Identitäten, Zugriffskontrolle sowie korrekter Handhabung verschiedener Arten der Authentifizierung. Der ASB zielt folglich zuerst darauf ab, sicherzustellen, dass ausschließlich berechtigte Personen Zugriff auf einen *Azure*-Mandanten und dessen Ressourcen nehmen können. Erst anschließend findet eine Betrachtung des Datenschutzes, des Bedrohungs- und Sicherheitsmanagements sowie der Reaktion auf Sicherheitsvorfälle statt.

Toroman und Janetscheck schließen an ihre Erwähnung von *Governance* ebenfalls eine Untersuchung des Identitätsmanagement und damit verbundener vitaler Punkte wie MFA (*Multi-Factor Authentication*), Zugriffskontrolle sowie vielseitiger Anmeldemethoden an. Daraufhin wird auch vorgeschlagen, die Netzwerk- und die Datensicherheit zu überprüfen, bevor eine Erläuterung der Tools *Azure Security Center* und *Azure Sentinel* sowie deren Verwendung beim Bedrohungs- und Sicherheitsmanagement und dem Einsatz von Gegenmaßnahmen stattfindet. Das CIS verfolgt zu Beginn einen ähnlichen Ansatz und steigt ebenfalls über das Identitäts- und Zugriffsmanagement in den *Hardening*-Prozess ein. Fortgesetzt wird dieser *Benchmark* jedoch mit der Sicherung von Speicherkonten und dem Schutz vor Bedrohungen mittels ASC. Die Netzwerksicherheit wird erst nach der Sicherheit von Datenbanken und dem Logging-Prozess betrachtet.

Alle drei Vorgehensweisen für das *Hardening* bieten einen vitalen Einstiegspunkt in die Sicherung einer *Azure*-Infrastruktur. In herkömmlichen IT-Infrastrukturen galt stets das Netzwerk und die darüber erwirkte Verbindung zum Internet als der Perimeter und der Einsatz von Sicherheitsmaßnahmen wurde an dieser Stelle begonnen. Während Netzwerksicherheit auch in der Cloud weiterhin eine Rolle spielt, gilt *Identity and Access Management* (IAM) als neue rigoros zu überprüfende Sicherheitsstufe in solchen Infrastrukturen [53]. Diesen Punkt heben die drei thematisierten Ansätze entsprechend in den Vordergrund.

5.1.2 Penetrationstest: ausgewählte Vorgehensweisen

Burrough bietet in seiner Vorgehensweise tiefe technische Einblicke darin, wie einzelne Services zu testen sind, welche Werkzeuge hierfür verwendet werden können sowie gegenteilig zu den Empfehlungen für das Vorgehen des Testers *Best Practices* für die Absicherung der getesteten Services. Weiterhin fokussiert Burrough seine Bemühungen auf die Services, welche nahezu immer anzutreffen sind und deckt daher die wahrscheinlichsten Testfälle mit detaillierten Durchführungshinweisen ab. Zudem bietet die Vorgehensweise Hintergrundwissen und Begründungen, um neben dem *Was* und *Wie* auch das *Warum* für einen Testfall zu beschreiben. Auf das zentrale Bedürfnis für eine lückenlose Definition des Testrahmens und die gesetzliche Absicherung des Tests legt Burrough außerdem großen Wert.

Umgekehrt ist anzumerken, dass Burrough zwar umfassend die wahrscheinlichsten Testfälle beschreibt, jedoch aufgrund der inhaltlichen Tiefe nicht jegliche Services gleichberechtigt darstellen kann. Zusätzlich handelt es sich bei *Pentesting Azure Applications* um ein Buch, welches als solches kein Industriestandard ist und dementsprechend mutmaßlich auch nicht als Standardpraxis anerkannt wird.

Das *Cloud Penetration Testing Playbook* der CSA bietet eine breite Einstiegsgrundlage für die Gestaltung eines Penetrationstests in einer universellen Cloud-Umgebung. Hierbei liegt der Fokus auf der organisatorischen Komponente eines Penetrationstests, sprich welche Fokuspunkte zu beachten und während der Planung und Durchführung eines Tests zu bearbeiten sind. Der Testrahmen eines Cloud-Penetrationstests wird anschaulich abgesteckt sowie Unterschiede zwischen Service-Modellen deutlich gemacht. Zudem empfiehlt die CSA eine Testmethodologie ähnlich der des BSI für Penetrationstests sowie Zielsetzungen in Form von zu prüfenden Schwachstellen-Kategorien gemäß des STRIDE-Modells von *Microsoft*.

Nachteilig für diese Vorgehensweise ist festzustellen, dass zwar eine große organisatorische Sorgfalt besteht, jedoch keine technischen Erklärungen und Anleitungen gegeben werden, wie und mit welchen Werkzeugen ein Testfall zu verifizieren ist. Somit bietet das *Cloud Penetration Testing Playbook* eine umfangreichere Fülle von Testfäl-

len und Schritten an, welche einen größeren Bezug zu bestimmten Schwachstellen-Kategorien herstellen. Die technische Realisierung dieser ist jedoch selbst zu erarbeiten beziehungsweise anderen Test-Frameworks zu entnehmen, beispielsweise denen des OWASP. Dies hat den Grund, dass die Vorgehensweise der CSA allgemeiner Natur ist und nicht spezifisch auf einen einzelnen *Provider* ausgerichtet wird.

Beide beschriebenen Vorgehensweisen widmen sich dem Thema Penetrationstest in der Cloud beziehungsweise in *Azure* auf eine andere Art und Weise. Burrough bietet einen Einblick in die technische Seite eines Tests, in der zwar auch organisatorische Elemente angesprochen werden, beispielsweise die Festlegung des Testrahmens sowie die gesetzliche und vertragliche Absicherung. Jedoch bietet *Pentesting Azure Applications* vor allem technische Beschreibungen, welche Aufschluss über die Durchführung der beschriebenen Testfälle geben. Zusätzlich bietet Burroughs Vorgehen eine nach Diensten geordnete Orientierung sowie beispielhafte Darstellungen der Ausführung von Tools und deren entsprechende Ergebnisse. Darüber hinaus werden *Best Practices* für die Behebung einzelner dargestellter Testfälle präsentiert.

Die CSA auf der anderen Seite widmet sich der vollständigen organisatorischen Planung eines Cloud-Penetrationstests. Hierzu wird ein hoher Stellenwert auf eine möglichst engmaschige Darstellung von Teilschritten gelegt, welche über die Phasen eines Penetrationstests absolviert werden sollten, von Elementen, welche bei der Festlegung des Testrahmens zu berücksichtigen sind bis hin zu einer Auflistung vielfältiger Testfälle, welche durchgeführt werden können. Im Gegensatz zu Burrough findet jedoch keine technische Erklärung der einzelnen durchzuführenden Tests statt. Vielmehr präsentiert die CSA eine Art Checkliste, welche eine umfassende Planung eines Cloud-Penetrationstests unter Berücksichtigung aller relevanten Aspekte ermöglichen soll. Zudem erfolgt die Organisation der Vorgehensweise im Gegensatz zu Burrough nicht in Dienste, sondern in Testphasen ähnlich dem BSI. Die Testfälle sind des Weiteren so ausgewählt, dass die Aspekte des STRIDE-Modells verifiziert werden können. Dies stellt einen klaren Unterschied dar, da die CSA eine allgemeine, von dem *Cloud Provider* unabhängige Vorgehensweise präsentiert.

5.1.3 Anwendbarkeit nach Service-Modellen: Aussagekraft

Die in Kapitel 4, Abschnitt 4.4 (siehe Seite 81) gestellten Empfehlungen für die Anwendung von Penetrationstests und *Hardening Checks* bei verschiedenen Service-Modellen sind insofern aussagekräftig, dass sie sich aus der theoretischen Erarbeitung dieser Arten von Sicherheitstests sowie bekannter Vorgehensweisen für diese begründen lassen. Hierbei ist zu beachten, dass für eine Einschätzung aus praktischer Sicht die notwendige Erfahrung nicht gegeben ist, um aus dieser Perspektive abwägen zu können.

Für den *Hardening Check* als Sicherheitstest ist zu berücksichtigen, dass dieser sich bei *Azure*, wie auch an den in dieser Arbeit erläuterten Vorgehensweisen zu erkennen ist, sich nicht spezifisch auf ein einzelnes Service-Modell ausrichtet. Vielmehr liegt das Ziel darin, eine universelle ganzheitliche Sicherheitsüberprüfung und im besten Fall -verbesserung bei einem *Azure*-Mandanten vorzunehmen, sodass eingesetzte Services oder Service-Modelle keine Rolle spielen oder separat voneinander berücksichtigt werden müssen. Dies kann natürlich trotzdem getan werden. Dementsprechend basieren die gefassten Einschätzungen für den *Hardening Check* bei den einzelnen Service-Modellen auf einer abwägenden Entscheidung zwischen dem potenziellen Nutzen und dem sich präsentierenden Aufwand beziehungsweise der Fülle an Konfigurationseinstellungen, welche überhaupt durch den Kunden getroffen werden können. Diese Einschätzungen sind keinesfalls eine stets zu wahrende Richtlinie, sondern gründen sich auf dem Verständnis der erfassten *Hardening*-Vorgehensweisen und Service-Modelle.

Bei dem Penetrationstest als Sicherheitstest verhält es sich gewissermaßen umgekehrt und eine Zuordnung einzelner Testarten zu einem Service-Modell kann getroffen werden. Hierbei ist zu berücksichtigen, dass diese Zuordnungen versuchen, grundlegende Rahmenbedingungen und Gegebenheiten der einzelnen Service-Modelle zu berücksichtigen und dementsprechend fundiert darauf einzugehen, welchen Nutzen die jeweilige Testart erbringen würde. Der *Black Box*-Test wurde deshalb kategorisch von vornherein ausgeschlossen, weil die geteilte Natur der Ressourcen einer Cloud das Testen einzelner IP-Adressen beispielsweise erschweren, da diese zwischen unterschiedlichen Kunden geteilt sein können. Selbstverständlich sind Szenarien denkbar, in denen ein solcher Test dennoch genutzt werden könnte. Jedoch fehlt an dieser Stelle ebenfalls die praktische Erfahrung dies weitergehend einzuschätzen. Ebenso verhält es sich mit der Einschätzung des Aufwandes und des potenziellen Nutzens bei *Grey Box*- und *White Box*-Tests in *Azure*. Diese basieren auf der Interpretation der Kundenverantwortung bei den einzelnen Service-Modellen und stellen keine starre Grenze dar, welche stets eingehalten werden muss.

5.2 Vergleich mit anderen Vorgehensweisen

5.2.1 Cloud Standards in Comparison - Di Giulio et al.

Di Giuolio et al. evaluieren den Nutzen und die Wirkung neuer Frameworks für Informationssicherheit im Cloud-Bereich. Hierbei soll herausgefunden werden, ob die vorgestellten Frameworks FedRAMP (*Federal Risk Authorization Management Program*), C5 (*Cloud Computing Compliance Control Catalogue*) sowie die bekannte ISO/IEC-Norm 27001 geeignet sind, um aktuelle Bedrohungen und Sicherheitsrisiken im *Cloud Computing* anzusprechen. Dies ist laut den Autoren notwendig, da die neu eingeführten

Standards einen großen Teil ihrer Grundlagen mit den traditionellen Normen wie der ISO 27001 teilen und deswegen der Mehrwert dieser ermittelt werden müsse. Das Ergebnis zeigt, dass die verglichenen neuen Standards keine tiefgreifenden Neuerungen im Bereich der Zertifikate für Informationssicherheit offenbaren [6].

Vorgegangen sind Di Giulio et al. hierbei so, dass zu Beginn die drei Standards bezüglich ihres Inhalts erläutert wurden. Anschließend sind deren Kontrollen einer Vergleichsvorgehensweise, der CSA *Cloud Control Matrix* (CSA CCM), gegenübergestellt worden, um daraufhin fehlende Kontrollen bezüglich ihrer Eignung für die Garantie der Cloud-Sicherheit zu analysieren. Unterstützt wurde dieses Vorgehen durch eine Betrachtung von bekannten Bedrohungen in der Cloud, basierend auf den *Treacherous Twelve - T12* der CSA. Das Ergebnis zeigt letztendlich, dass alle drei eingeführten und betrachteten Standards gegenüber der Bedrohungslage in der Cloud und den Fokuspunkten der CSA CCM unzureichende Sicherheit bieten. In Kombination können die drei Vorgehensweisen jedoch eine Art Synergie erzeugen und einen Großteil der durch die T12 dargestellten Bedrohungen abdecken. [6].

Das Vorgehen dieser Bachelorarbeit überschneidet sich mit Di Giulio et al. insofern, dass ebenfalls bekannte Bedrohungen und Angriffsvektoren ermittelt wurden, um einen Kontext der Bedrohungslage in der Cloud zu bieten. Des Weiteren wurden unterschiedliche Vorgehensweisen für zwei Arten von Sicherheitstests betrachtet und hinsichtlich ihrer Eignung für die unterschiedlichen Service-Modelle in der Cloud, spezifisch *Azure*, verglichen und empfohlen. Hier gehen Di Giulio et al. einen anderen Weg, indem sie verschiedene Industriestandards für Informationssicherheit gegenüberstellen, um deren Einfluss und Eignung hinsichtlich der aktuellen Bedrohungslandschaft in der Cloud allgemein herauszuarbeiten. Weiterhin liegt der Fokus der Ergebnisse auf der Ermittlung, inwiefern etablierte Sicherheitsstandards tatsächlich einen sicheren Anhaltspunkt für eine Bekämpfung der prominenten Bedrohungen für die Cloud liefern. Umgekehrt präsentiert diese Arbeit eine Übersicht über bestehende und potenziell nutzbare Vorgehensweisen, das Sicherheitsniveau eines *Azure*-Mandanten beziehungsweise einer *Azure*-Infrastruktur zu testen und zu verbessern sowie deren Empfehlung, insbesondere bezüglich der detaillierten technischen Realisierungen und Hintergründe spezifisch für *Azure*.

5.2.2 A security evaluation framework for cloud security auditing - Rizvi et al.

Bei Rizvi et al. handelt es sich um eine Betrachtung der Vielzahl an verfügbaren *Cloud Providern*, in Bezug auf deren Eignung hinsichtlich der Sicherheitsbedürfnisse eines Kunden. Ziel dieses Werkes ist die Bereitstellung eines Frameworks, welches bei der Entscheidung für den richtigen *Provider* unterstützend angewandt werden kann. Hierbei sollen die Stärken der einzelnen Anbieter gegenübergestellt und vergleichend betrach-

tet werden, um deren unterschiedliche Vorteile für verschiedene Sicherheitsaspekte herauszuarbeiten. Aus den daraus resultierenden Erkenntnissen wird eine Vorgehensweise zur bedarfsgerechten Identifikation des bestgeeigneten *Cloud Service Providers* (CSP) abgeleitet [46].

Die Vorgehensweise von Rizvi et al. beginnt mit einer Erläuterung des erdachten Konzepts für die Bewertung der Sicherheit eines CSP. Dieses stützt sich auf berechnete *Security Scores*, welche auf unterschiedliche Art und Weise erreicht werden können. Diese werden nach der Darstellung des Konzepts erarbeitet. Anschließend werden verschiedene Wege aufgezeigt, lineare Gleichungen für die Berechnung dieser Punktzahlen zu verwenden, bevor das Bewertungskonzept für die Ermittlung der *Security Scores* unterschiedlicher Fallstudien genutzt wird, um dessen Funktionsweise zu demonstrieren und den Nutzen zu beweisen. Das Ergebnis besteht aus einem Bewertungskonzept, welches auf der numerischen Gewichtung der Kriterien *Transparenz* und *Prävention maliziöser Insider* basiert, anhand derer in verschiedenen Szenarien zwei hypothetische CSP gegeneinander verglichen werden [46].

Rizvi et al. versucht, mithilfe der Mathematik eine vereinheitlichte Beweisbarkeit und Mittelbarkeit für die Entscheidungskriterien bei der Auswahl eines CSP zu realisieren. Hierbei liegt das Ziel darin, anhand der spezifischen Sicherheitsbedürfnisse einer Organisation genau den CSP auszuwählen, welcher diese Bedürfnisse am besten erfüllen kann. Der Unterschied zu dieser Arbeit liegt darin, dass Rizvi et al. den Fokus auf die allgemeine Erfüllung der Sicherheitsbedürfnisse vonseiten des CSPs legen, ohne das die betreffende Organisation beziehungsweise der Kunde von sich aus etwas unternehmen muss. Umgekehrt wird in dieser Bachelorarbeit betrachtet, inwiefern der Kunde selbst für eine optimale Erfüllung der eigenen Sicherheitsbedürfnisse über Sicherheitstests und die daraus resultierende Optimierung des Sicherheitsniveaus sorgen kann, anhand von Einschätzungen über den Einsatz bestimmter Vorgehensweisen und Überprüfungen in unterschiedlichen Verwendungsszenarien von *Microsoft Azure*. Dies stellt einen weiteren Unterschied zu Rizvi et al. dar, da das dort beschriebene Vorgehen allgemeiner Natur ist und theoretische CSP miteinander verglichen werden, während diese Arbeit versucht, die spezifischen Gegebenheiten und technischen Fokuspunkte von *Azure* zu berücksichtigen.

5.2.3 Analysis of Cloud Security Controls in AWS, Azure and Google Cloud - Sailakshmi

Sailakshmi strebt in seiner Vorgehensweise den Vergleich und die Gegenüberstellung der Top 20 Sicherheitskontrollen der CSA für Cloudsicherheit mit den jeweiligen Sicherheitskontrollen von AWS, *Azure* und der *Google Cloud Platform* (GCP) an. Das Ziel liegt darin, eine Entscheidungshilfe für Organisationen und Cloud-Nutzer zu liefern, anhand derer zweckdienliche Informationen über die verschiedenen Sicherheitskontrollen und deren Funktion erlangt werden können [47].

Begonnen hat Sailakshmi mit einer Betrachtung von relevanten Sicherheitsverlusten in den einzelnen Clouds, um einen Überblick über bestehende Risiken zu erhalten. Fortgefahren wurde mit der Feststellung der verschiedenen Hard- und Softwareumgebungen, Kontrollprinzipien, -dimensionen und Verantwortlichkeiten für jeden der einzelnen *Cloud Provider*. Zuletzt werden die Sicherheitskontrollen der CSA erläutert und die jeweiligen Sicherheitsfeatures in AWS, *Azure* und GCP ermittelt, welche diese Kontrollen erfüllen. Das Ergebnis bescheinigt den drei Providern das Vorhandensein der notwendigen Sicherheitsmaßnahmen, um den Kontrollen der CSA gerecht zu werden. Jedoch wird angemerkt, dass die Konfiguration der einzelnen Services umfassend dokumentiert ist und durch deren komplexe Natur schnell unzureichend sicher sein kann [47].

Die primäre Differenz zwischen Sailakshmi und dieser Bachelorarbeit liegt darin, dass Sailakshmi den Fokus auf keine bestimmte Cloud, sondern auf die drei großen *Cloud Provider Amazon*, *Microsoft* und *Google* legt, um eine möglichst umfassende, universelle Orientierung zu bieten, während diese Arbeit ausschließlich *Microsoft Azure* betrachtet. Die beiden Vorgehen gleichen sich darin, dass der Fokus auf der Ermittlung von Möglichkeiten zur Aufrechterhaltung eines angemessenen Sicherheitsniveaus in der Cloud liegt. Sailakshmi setzt hierbei jedoch ausschließlich auf die Sicherheitsvorkehrungen, welche die einzelnen *Cloud Provider* selbst anbieten. Dies wird in dieser Arbeit spezifisch für *Azure* auch berücksichtigt, jedoch besteht der Kern der Arbeit aus der Ermittlung von Testvorgehensweisen, welche von einer Organisation selbst in Auftrag gegeben werden können, um die erreichte Sicherheit auch bedarfsgerecht zu verifizieren und zu überprüfen.

5.3 Ausblick

Aufbauend auf den Ergebnissen dieser Arbeit, welche theoretischer Natur sind, wäre für die Zukunft denkbar, dies mit praktischen Erfahrungen in diesem Bereich zu ergänzen und getroffene Empfehlungen zu verfeinern. Auf diese Weise wäre ebenfalls eine Erprobung der einzelnen ausgewählten Vorgehensweisen auf ihren tatsächlichen Nutzen, den damit verbundenen Aufwand und die letzten Endes dadurch zu erreichende Sicherheit möglich.

Weiterhin ist es vorstellbar, eine spezifische Vorgehensweise für einen *Hardening Check* herauszugreifen und zu versuchen, diese zu automatisieren. Hierfür könnten auch weitere Standards und Vorgehensweisen gesucht und in der hier dargestellten Art und Weise betrachtet werden. Bei den Penetrationstests könnte versucht werden, Plugins für bekannte Schwachstellenscanner zu testen und auf ihre Zuverlässigkeit hin zu bewerten. Hierbei wäre auch die Entwicklung eines eigenen Schwachstellenscanners denkbar. Auch eine simple Durchführung eines Penetrationstests in *Azure* um die in dieser Arbeit beschriebenen Vorgehensweisen hinsichtlich ihres Nutzens zu evaluieren könnte Bestandteil zukünftiger Arbeiten in diesem Bereich sein.

Ebenso wäre eine Ausweitung auf weitere Cloud-Anbieter denkbar, um beispielsweise für AWS funktionierende Vorgehensweisen und Empfehlungen zu identifizieren. Dies wäre beispielsweise deswegen interessant, da AWS wie bereits in Kapitel 1 (siehe Seite 1) erläutert, als Marktführer im Bereich Cloud gilt. Dies ließe sich ebenfalls für jeden weiteren *Cloud Provider* durchführen.

5.4 Fazit

Die Erfassung und Erarbeitung unterschiedlicher Vorgehensweisen für Penetrationstests und *Hardening Checks* in *Azure*-Infrastrukturen hat gezeigt, dass mehr als ein Ansatz existiert, die Sicherheit eines *Azure*-Mandanten und dessen Ressourcen und Services zu evaluieren und gegebenenfalls zu verbessern. Hierbei sind die Differenzen zwischen der Verantwortung des *Cloud Providers* und des Kunden stets zu berücksichtigen, um abwägen zu können, in welchem Umfang und ob der jeweilige Sicherheitstest sinnvoll ist und durchgeführt werden kann.

Ebenso ist klar erkennbar, dass der Einsatz einer Organisation in *Azure* einer Reihe von Bedrohungen ausgesetzt ist, welchen mit einem sorgfältig ausgewählten und geplanten Sicherheitstest begegnet werden kann, um etwaige Risiken für die Sicherheit beispielsweise sensibler Daten zu minimieren. In Abhängigkeit der einzelnen Service-Modelle einer Cloud sind die einzelnen Vorgehensweisen und Sicherheitstests unterschiedlich empfehlenswert. Jedoch kristallisiert sich heraus, dass ein Penetrationstest allgemein besser anwendbar ist, als ein *Hardening Check*, da die Service-Modelle SaaS und PaaS vergleichsweise wenig Ansatzpunkte für einen derartigen Sicherheitstest bieten.

Letztendlich entsprechen die aus dieser Arbeit gezogenen Erkenntnisse dem derzeitigen theoretischen Wissensstand, sind jedoch keinesfalls umfassend erschöpft. Die hier begonnenen Untersuchungen müssen, vor allem praktisch, weiter fortgesetzt werden, um die Anwendbarkeit von Sicherheitstests in *Azure* zu untersuchen und weitere Vorgehensweisen, Methodiken und Anwendungsgebiete zu erschließen.

Anhang A: Ergebnisse

A.1 CIS Benchmark-Beispiel

Folgender Textabschnitt stellt eine exemplarische Empfehlung aus dem Bereich Indentitäts- und Zugriffsmanagement aus dem CIS *Microsoft Azure Foundations Benchmark* dar[6].

1.1 Ensure that multi-factor authentication is enabled for all privileged users (Manual)

Profile Applicability:

- **Level 1**

Description:
Enable multi-factor authentication for all user credentials who have write access to Azure resources. These include roles like

- Service Co-Administrators
- Subscription Owners
- Contributors

Rationale:
Multi-factor authentication requires an individual to present a minimum of two separate forms of authentication before access is granted. Multi-factor authentication provides additional assurance that the individual attempting to gain access is who they claim to be. With multi-factor authentication, an attacker would need to compromise at least two different authentication mechanisms, increasing the difficulty of compromise and thus reducing the risk.

Impact:
Users would require two forms of authentication before any action is granted. Also, this requires an overhead for managing dual forms of authentication.

[6] O.A.: **CIS Microsoft Azure Foundations Benchmark v1.3.0**, CIS, 2021, Zugriff 04.08.2021, URL `https://www.cisecurity.org/benchmark/azure/`

Audit:

From Azure Console

1. Go to `Azure Active Directory`
2. Go to `Users`
3. Go to `All Users`
4. Click on `Multi-Factor Authentication` button on the top bar
5. Ensure that `MULTI-FACTOR AUTH STATUS` is `Enabled` for all users who are `Service Co-Administrators` OR `Owners` OR `Contributors`

Microsoft Graph API
For Every Subscription, For Every Tenant
Step 1: Identify Users with Administrative Access

1. List All Users Using Microsoft Graph API:

   ```
   GET https://graph.microsoft.com/v1.0/users
   ```

   Capture `id` and corresponding `userPrincipalName` (`$uid`,`$userPrincipalName`)

2. List all Role Definitions Using Azure management API:

   ```
   https://management.azure.com/subscriptions/
   :subscriptionId/providers/Microsoft.Authorization/
   roleDefinitions?api-version=2017-05-01
   ```

   Capture Role Definition IDs/Name (`$name`) and role names (`$properties/roleName`) where „`properties/roleName`“ contains (`Owner` or `*contributor` or `admin`)

3. List All Role Assignments (Mappings `$A.uid` to `$B.name`) Using Azure Management API:

   ```
   GET https://management.azure.com/subscriptions/
   :subscriptionId/providers/Microsoft.Authorization/
   roleassignments?api-version=2017-10-01-preview
   ```

   Find all administrative roles (`$B.name`) in `"Properties/roleDefinitionId"` mapped with user ids (`$A.id`) in `"Properties/principalId"` where `"Properties/principalType" == "User"`

4. Now Match (`$CProperties/principalId`) with `$A.uid` and get `$A.userPrincipalName` save this as `D.userPrincipalName`

Step 2: Run MSOL Powershell command:

```
Get-MsolUser -All | where {\$\_.StrongAuthentication
Methods.Count -eq 0} | Select-Object -Property
UserPrincipalName
```

If the output contains any of the `$D.userPrincipalName`, then this recommendation is non-compliant.
Please note that at this point of time, there is no API/CLI mechanism available to programmatically conduct security assessment for this recommendation. Only option is MSOL

Remediation:
Follow Microsoft Azure documentation and setup multi-factor authentication in your environment.
`https://docs.microsoft.com/en-us/azure/active-directory/authentication/tutorial-enable-azure-mfa`

Default Value:
By default, multi-factor authentication is disabled for all users.

References:

1. `https://docs.microsoft.com/en-us/azure/multi-factor-authentication/multi-factor-authentication`
2. `https://stackoverflow.com/questions/41156206/azure-active-directory-premium-mfa-attributes-via-graph-api`
3. `https://docs.microsoft.com/en-us/azure/security/benchmarks/security-controls-v2-identity-management#im-4-use-strong-authentication-controls-for-all-azure-\newlineactive-directory-based-access`

CIS Controls:
Version 7

4.5 Use Multifactor Authentication For All Administrative Access
Use multi-factor authentication and encrypted channels for all administrative account access.

Literaturverzeichnis

[1] BRANDT, Mathias: **Amazon ist die Nummer 1 in der Cloud**, Online-Artikel, statista, 2020, Zugriff 09.06.2021, URL https://de.statista.com/infografik/20802/weltweiter-marktanteil-von-cloud-infrastruktur-dienstleistern/.

[2] BROOK, Jon-Michael C.: **Top Threats to Cloud Computing - The Egregious 11**, CSA, 2020, Zugriff 29.07.2021, URL https://cloudsecurityalliance.org/artifacts/top-threats-to-cloud-computing-egregious-eleven/.

[3] BURROUGH, Matt: **Pentesting Azure Applications**, San Francisco, No Starch Press, 2018, ISBN-13 9781593278632, URL https://lccn.loc.gov/2017051237.

[4] CHEN, Lei et al.: **Security, Privacy and Digital Forensics in the Cloud**, John Wiley & Sons Singapore, 2019, Zugriff 29.07.2021, URL https://doi.org/10.1002/9781119053385.

[5] COPPOLINO, Luigi et al.: **Cloud security: Emerging threats and current solutions**, In: ELSEVIER LTD. (Hrsg.) *Computers & Electrical Engineering 59*, S.126-140, 2017, Zugriff 19.07.2021, URL https://doi.org/10.1016/j.compeleceng.2016.03.004.

[6] DI GIULIO, Carlo et al.: **Cloud Standards in Comparison**, In: IEEE (Hrsg.) *2017 IEEE 10th International Conference on Cloud Computing*, S.50-57, 2017, Zugriff 03.09.2021, URL https://doi.org/10.1109/CLOUD.2017.16.

[7] GETSIN, Alexander et al.: **Cloud Penetration Testing Playbook**, Cloud Security Alliance (CSA), 2019, Zugriff 11.08.2021, URL https://cloudsecurityalliance.org/artifacts/cloud-penetration-testing-playbook/.

[8] KARLSTETTER, Florian: **Was ist On-Premises?**, Online-Artikel, Cloudcomputing Insider, 2017, Zugriff 17.06.2021, URL https://www.cloudcomputing-insider.de/was-ist-on-premises-a-623402/.

[9] KARTHIKEYAN, Shijimol A.: **Demystifying the Azure Well-Architected Framework**, Apress Berkeley (CA), 2021, Zugriff 05.08.2021, URL https://doi.org/10.1007/978-1-4842-7119-3.

[10] LUBER, Stefan Dipl.-Ing.; KARLSTETTER, Florian: **Was ist ein Workload?**, Online-Artikel, Cloudcomputing Insider, 2020, Zugriff 25.06.2021, URL https://www.cloudcomputing-insider.de/was-ist-ein-workload-a-901115/.

[11] LUBER, Stefan Dipl.-Ing.; KARLSTETTER, Florian: **Was ist Microsoft Azure?**, Online-Artikel, Cloudcomputing Insider, 2017, Zugriff 02.06.2021, URL https://www.cloudcomputing-insider.de/was-ist-microsoft-azure-a-667912/.

[12] LUBER, Stefan Dipl.-Ing.; SCHMITZ, Peter: **Was ist ein Penetrationstest?**, Online-Artikel, Security Insider, 2017, Zugriff 02.07.2021, URL https://www.security-insider.de/was-ist-ein-penetrationstest-a-667683/.

[13] MACHIRAJU, Suren; GAURAV, Suraj: **Hardening Azure Applications**, Apress, 2019, Zugriff 02.08.2021, URL https://doi.org/10.1007/978-1-4842-4188-2.

[14] MELL, Peter; GRANCE, Timothy: **The NIST Definition of Cloud Computing**, In: COMPUTER SECURITY DIVISION, INFORMATION TECHNOLOGY LABORATORY, NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (Hrsg.) *Special Publication 800-145*, 2011, Zugriff 10.06.2021, URL https://csrc.nist.gov/publications/detail/sp/800-145/final.

[15] NAYBZADEH, Milan: **Rechtliche Vorgaben zur IT-Sicherheit und IT-Compliance**, In: ADACOR (Hrsg.) *Adacor Blog: Impulse für Ihr digitales Business*, Blog-Artikel, 2021, Zugriff 09.08.2021, URL https://blog.adacor.com/gesetzliche-anforderungen-it-compliance_1055.html.

[16] O.A.: **Angriffsvektor**, In: ISTQB (Hrsg.) *Webseite: ISTQB Glossary*, 2016, Zugriff 20.07.2021, URL https://glossary.istqb.org/de/term/angriffsvektor.

[17] O.A.: **Audit Microsoft Azure in Nessus**, In: TENABLE (Hrsg.) *Webseite: tenable Documentation*, o.J., Zugriff 07.08.2021, URL https://docs.tenable.com/integrations/Microsoft/Azure/Content/audit_azure_nessus.htm.

[18] O.A.: **Azure-Abonnements und -Verwaltungsgruppen**, In. MICROSOFT (Hrsg.) *Microsoft Docs: Learn*, o.J., Zugriff 13.07.2021, URL https://docs.microsoft.com/de-de/learn/modules/azure-architecture-fundamentals/management-groups-subscriptions.

[19] O.A.: **Azure security benchmark introduction**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Azure*, 2021, Zugriff 04.08.2021, URL https://docs.microsoft.com/en-us/security/benchmark/azure/introduction.

[20] O.A.: **Azure security best practices**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Azure*, 2020, Zugriff 07.08.2021, URL https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/secure/security-top-10.

[21] O.A.: **Azure Stack**, In: MICROSOFT (Hrsg.) *Microsoft Azure: Webseite*, o.J., Zugriff 03.09.2021, URL https://azure.microsoft.com/de-de/overview/azure-stack/.

[22] O.A.: **CIS Benchmarks FAQ**, In: CENTER FOR INTERNET SECURITY (Hrsg.) *Webseite: CIS Benchmarks - CIS Benchmarks FAQ*, o.J., Zugriff 16.08.2021, URL https://www.cisecurity.org/cis-benchmarks/cis-benchmarks-faq/.

[23] O.A.: **CIS Microsoft Azure Foundations Benchmark v1.3.0**, CIS, 2021, Zugriff 04.08.2021, URL https://www.cisecurity.org/benchmark/azure/.

[24] O.A.: **Cloud Computing - Was steckt dahinter?**, In: IONOS (Hrsg.) *Webseite: Digital Guide*, 2020, Zugriff 02.06.2021, URL https://www.ionos.de/digitalguide/server/knowhow/cloud-computing-definition-erklaerung-geschichte/.

[25] O.A.: **Datenschutz-Grundverordnung (DSGVO)**, EU-Richtlinie, 2016, URL https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:02016R0679-20160504.

[26] O.A.: **Ein Praxis-Leitfaden für IS-Penetrationstests**, Publikation, BSI, 2016, Zugriff 05.07.2021, URL https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Sicherheitsberatung/Pentest_Webcheck/Leitfaden_Penetrationstest.pdf.

[27] O.A.: **Erstellen und Verwalten von Benutzern**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Learn*, o.J., Zugriff 15.07.2021, URL https://docs.microsoft.com/de-de/learn/modules/manage-users-and-groups-in-aad/3-users.

[28] O.A.: **Erste Schritte mit Azure-Konten**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Learn*, o.J., Zugriff 13.07.2021, URL https://docs.microsoft.com/de-de/learn/modules/intro-to-azure-fundamentals/get-started-with-azure-accounts.

[29] O.A.: **Hypervisor: Mittler für die Virtualisierung**, In: IONOS (Hrsg.) *Webseite: Digital Guide*, 2020, Zugriff 07.07.2021, URL https://www.ionos.de/digitalguide/server/knowhow/was-ist-ein-hypervisor/.

[30] O.A.: **Introduction to regulatory compliance**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Azure*, 2019, Zugriff 09.08.2021, URL https://docs.microsoft.com/en-us/azure/cloud-adoption-framework/govern/policy-compliance/regulatory-compliance.

[31] O.A.: **Microsoft Azure Audit Compliance Reference**, In: TENABLE (Hrsg.) *Webseite: tenable Documentation*, o.J., Zugriff 07.08.2021, URL https://docs.tenable.com/nessus/compliancechecksreference/Content/MSAzureAuditComplianceFileReference.htm.

[32] O.A.: **Nessus Demo**, In: NESSUS (Hrsg.) *Webseite: nessus professional Demo: Live Results*, o.J., Zugriff 05.09.2021, URL https://www.tenable.com/nessus-demo.

[33] O.A.: **Overview of the Azure Security Benchmark (V2)**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Azure*, 2021, Zugriff 04.08.2021, URL https://docs.microsoft.com/en-us/security/benchmark/azure/overview.

[34] O.A.: **Penetration testing**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Azure*, 2021, Zugriff 10.08.2021, URL https://docs.microsoft.com/en-us/azure/security/fundamentals/pen-testing.

[35] O.A.: **Penetrationstests - Rules of Engagement**, In: MICROSOFT (Hrsg.) *Webseite: Microsoft Security Response Center*, o.J., Zugriff 18.08.2021, URL https://www.microsoft.com/de-de/msrc/pentest-rules-of-engagement?rtc=1.

[36] O.A.: **Security Controls V2: Network Security**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Azure*, 2021, Zugriff 05.09.2021, URL https://docs.microsoft.com/en-us/security/benchmark/azure/security-controls-v2-network-security.

[37] O.A.: **Strafgesetzbuch (StGB)**, Gesetz der Bundesrepublik Deutschland, o.J., Zugriff 09.08.2021, URL https://www.gesetze-im-internet.de/stgb/.

[38] O.A.: **Studie - Durchführungskonzept für Penetrationstests**, Publikation, BSI, 2003, Zugriff 02.07.2021, URL https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Studien/Penetrationstest/penetrationstest.html.

[39] O.A.: **Telemediengesetz (TMG)**, Gesetz der Bundesrepublik Deutschland, o.J., Zugriff 09.08.2021, URL https://www.gesetze-im-internet.de/tmg/index.html#BJNR017910007BJNE001301118.

[40] O.A.: **Tour durch die Azure-Dienste**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Learn*, o.J., Zugriff 09.07.2021, URL https://docs.microsoft.com/de-de/learn/modules/intro-to-azure-fundamentals/tour-of-azure-services.

[41] O.A.: **Was ist Azure?**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Learn*, o.J., Zugriff 03.07.2021, URL https://docs.microsoft.com/de-de/learn/modules/intro-to-azure-fundamentals/what-is-microsoft-azure.

[42] O.A.: **Was ist Azure Active Directory?**, In: MICROSOFT (Hrsg.) *Microsoft Docs: Learn*, o.J., Zugriff 13.07.2021, URL https://docs.microsoft.com/de-de/learn/modules/manage-users-and-groups-in-aad/2-create-aad.

[43] O.A.: **Was ist Cloudbursting?**, In: MICROSOFT (Hrsg.) *Mircosoft Azure-Webseite: Übersicht*, o.J., Zugriff 18.06.2021, URL https://azure.microsoft.com/de-de/overview/what-is-cloud-bursting/.

[44] REDAKTION COMPUTERWEEKLY.DE; TECHTARGET: **Definition: Multi-Tenancy (Mandantenfähigkeit)**, In: WHATIS.COM (Hrsg.) *Computer-Glossar mit über 10.000 Computerbegriffen*, o.J., Zugriff 10.06.2021, URL https://whatis.techtarget.com/de/definition/Multi-Tenancy-Mandantenfaehigkeit.

[45] RIDGWAY, Ben et al.: **Security Best Practices for Windows Azure Solutions**, Microsoft, 2014, Zugriff 02.08.2021, URL https://www.kiloroot.com/wp-content/uploads/2014/05/SecurityBestPracticesForWindowsAzureSolutionsFeb2014.pdf.

[46] RIZVI, Syed et al.: **A security evaluation framework for cloud security auditing**, In: *J Supercomput* 74, S.5744-5796, Springer, 2018, Zugriff https://doi.org/10.1007/s11227-017-2055-1.

[47] SAILAKSHMI, Vyshnavi: **Analysis of Cloud Security Controls in AWS, Azure and Google Cloud**, In: *Culminating Projects in Information Assurance* 112, 2021, Zugriff 04.09.2021, URL https://repository.stcloudstate.edu/msia_etds/112.

[48] SCARFONE, Karen et al.: **Technical Guide to Information Security Testing and Assessment**, In: COMPUTER SECURITY DIVISION, INFORMATION TECHNOLOGOY LABORATORY, NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY (Hrsg.) *Special Publication SP 800-115*, 2008, Zugriff 02.07.2021, URL https://csrc.nist.gov/publications/detail/sp/800-115/final.

[49] SHAH, Sugandh; MEHTRE, B.M.: **An overview of vulnerability assessment and penetration testing techniques**, In: *Journal of Computer Virology and Hacking Techniques 11*, S.27-49, 2015, Zugriff 06.07.2021, URL https://doi.org/10.1007/s11416-014-0231-x.

[50] SROCKE, Dirk, KARLSTETTER, Florian: **Was ist eine Cloud-Infrastruktur?**, Online-Artikel, Cloudcomputing Insider, 2018, Zugriff 18.06.2021, URL https://www.cloudcomputing-insider.de/was-ist-eine-cloud-infrastruktur-a-732116/.

[51] STATISTA RESEARCH DEPARTMENT: **Anteil der Cloud-nutzenden Unternehmen in Deutschland, die kostenfplichtige Dienste für folgende Zwecke nutzen im Jahr 2018**, Statistik, statista, 2021, Zugriff 09.06.2021, URL https://de.statista.com/statistik/daten/studie/381830/umfrage/einsatzzwecke-von-cloud-computing-in-unternehmen-in-deutschland/.

[52] TOROMAN, Mustafa; JANETSCHECK, Tom: **Mastering Azure Security**, Packt Publishing, 2020, Zugriff 02.08.2021, URL https://www.packtpub.com/product/mastering-azure-security/9781839218996.

[53] UNIT 42: **Cloud Threat Report 2H 2020**, Paloalto Networks, 2020, Zugriff 05.08.2021, URL https://www.paloaltonetworks.com/prisma/unit42-cloud-threat-research.

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich meine Arbeit selbstständig verfasst, keine anderen als die angegebenen Quellen und Hilfsmittel benutzt und die Arbeit noch nicht anderweitig für Prüfungszwecke vorgelegt habe.

Stellen, die wörtlich oder sinngemäß aus Quellen entnommen wurden, sind als solche kenntlich gemacht.

Mittweida, 6. September 2021