Hochschule Wismar
University of Applied Sciences Technology, Business and Design
Fakultät für Ingenieurwissenschaften
Bereich Elektrotechnik und Informatik

Master-Thesis

Möglichkeiten und Grenzen der automatisierten Detektion und Mitigation von Schwachstellen in Cloud-Konfigurationen mittels Cloud Security Posture Management

Eingereicht am: 26. Juni 2023

von: Andreas Gollwitzer

Betreuer: Prof. Dr. Nils Gruschka
Zweitbetreuerin: Prof. Dr. Antje Raab-Düsterhöft

Aufgabenstellung

Unternehmen setzen zunehmend auf Cloud-Infrastrukturen, um sich im globalen Wettbewerb zu behaupten. Agile Arbeitsmethoden, mit automatisierten Entwicklungsprozessen und virtualisierter Infrastruktur, werden zum Standard. Traditionelle Vorgehensweisen zur Prüfung der Informationssicherheit, die vor allem auf zeitlich geplanten und manuellen Auditierungsverfahren beruhen, werden dieser Dynamik nicht gerecht. Studien belegen bereits eine signifikante Zunahme an Sicherheitsrisiken, die auf fehlerhafte Cloud-Konfigurationen zurückzuführen sind. Das Marktforschungsunternehmen Gartner Inc. hat in diesem Kontext den Begriff *Cloud Security Posture Management* geprägt.

Die Masterthesis fokussiert auf die Untersuchung von Ansätzen zur automatisierten Detektion potentieller Schwachstellen in der kundenseitigen Konfiguration der Cloud-Infrastruktur. Die Fähigkeiten zur Detektion über die Programmierschnittstelle des Cloud-Anbieters oder anderer marktüblicher Techniken zur automatisierten Verwaltung von Cloud-Infrastrukturen, sind zu untersuchen. Eine Automatisierung der Sicherheitsprüfung verlangt ebenfalls die Definition und den Aufbau maschinenlesbarer Sicherheitsrichtlinien. In der Thesis sind die Anforderungen hierzu aufzuzeigen, sowie eine strukturelle Vorlage zu erstellen. Eine Literaturrecherche bewährter und empfohlener Cloud-Sicherheitsvorgaben, sogenannter Best-Practises, ist durchzuführen. Die Ergebnisse sind zu bewerten und idealerweise auch als Basis für automatisierbare Sicherheitsrichtlinien nutzbar.

Im Vorfeld einer prototypischen Umsetzung sind gegebenenfalls alternative Ansätze zu skizzieren und einzuordnen. Sowohl die Untersuchung der Programmierschnittstelle, als auch die prototypische Umsetzung ausgewählter Szenarien, sind anhand eines Cloud-Anbieters auszuführen.

Darauf aufbauend ist der perspektivische Mehrwert einer automatisierten Sicherheitsauditierung zur Compliance, anhand eines ausgewählten Sicherheitsstandards, darzustellen.

Kurzfassung

Der zunehmende Einsatz von Cloud-Infrastrukturen kann Wettbewerbsvorteile und Flexibilität für Unternehmen schaffen. Aktuelle Studien belegen jedoch, dass ein signifikanter Anteil an untersuchten Cloud-Konfigurationen teils gravierende Sicherheitsmängel aufweist.

Die Arbeit untersucht die Möglichkeiten und Grenzen Sicherheitsrichtlinien zu kodieren und über automatisiere Verfahren vorliegende Cloud-Konfigurationen hinsichtlich bestehender Sicherheitsmängeln zu auditieren. Zur Behebung identifizierter Sicherheitsmängel sollen idealerweise Maßnahmen automatisiert vorgeschlagen werden. Das Marktforschungsunternehmen Gartner Inc. hat den Begriff Cloud Security Posture Management (CSPM) in diesem Kontext geprägt.

Die Analyse des Application Programming Interface (API) eines Cloud-Anbieters (AWS), die als Informationsbasis für die automatisierte Sicherheitsprüfung dienen kann, zeigt deutlich auf, dass ein umfangreicher Zugriff auf sicherheitsrelevante Konfigurationsinformationen der Cloud-Infrastruktur machbar ist. Werden Cloud-Konfigurationen auf Basis von Infrastructure as Code (IaC) Daten analysiert, kann eine wesentliche Grenze des API-basierten Ansatzes ausgeschlossen werden.

Die Arbeit diskutiert und bewertet unterschiedliche technische Implementierungsalternativen. Die prototypische Umsetzung auf Basis eines Security-Offloaders zeigt eine hohe funktionale Abdeckung der erarbeiteten funktionalen Anforderungen. Der Umsetzungsaufwand je Sicherheitsrichtlinie ist jedoch relativ hoch mit circa zwei bis zu acht Stunden je Richtlinie und verlangt eine intensive Einarbeitung. Zur Steigerung der Effizienz ist es empfehlenswert Best-Practises einzusetzen. Die Arbeit hat unterschiedliche Quellen eingehend untersucht und bewertet. Sowohl Sicherheitsempfehlungen der Cloud-Anbieter als auch kommerzieller Sicherheitsunternehmen weisen die besten Ergebnisse auf.

Der herausgearbeitete Wertbeitrag von CSPM zur Verbesserung der Konformität zu Sicherheitsstandards wie dem BSI IT-Grundschutz Kompendium oder den Normen von NIST und CIS kann eine Investition in die Methodik vor allem dann rechtfertigen, wenn eine hohe Änderungshäufigkeit der Cloud-Infrastruktur zu erwarten ist.

Abstract

Potentials and Limits of Automated Vulnerability Detection and Mitigation in Cloud Configurations based on Cloud Security Posture Management

Enterprises increase their Cloud-infrastructure usage to gain competitive advantages and improve their flexibility. But recent studies show that a significant percentage of Cloud-configurations are suffering from partly sever information security weaknesses.

The thesis researches the potentials and limits to codify security policies and thus automate the security audit of existing cloud configurations regarding the existence of security vulnerabilities. Identified weaknesses shall be mitigated by automating remediation suggestions. The technology research firm Gartner Inc. created the term CSPM in this context.

The results of the executed API analysis of one Cloud Service Provider (CSP) (AWS), that can serve as an information source for the security automation, show a broad and detailed access to security-relevant cloud-configuration data. When using Infrastructure as Code (IaC) as source for cloud-configuration a major weakness of the API-based approach can be eliminated.

Different technical implementation approaches get evaluated and rated. The developed prototype is based on a security-offloader framework and shows an overall good functional fit to the defined functional requirements in this document. But the efforts are rated rather high with two up to eight hours pure coding per defined security policy and requires intense preparation. To improve efficiency it is thus strongly recommended to leverage best-practices. The thesis assessed and rated different sources. Both commercial security advisers and security advice from CSP offer best results.

The value add of CSPM to improve security compliance with standards like BSI IT-Grundschutz compendium or security norms from NIST and CIS can justify an investment especially when a high rate of change to cloud-infrastructure is expected.

Inhaltsverzeichnis

1 Einleitung

1.1 Ausgangslage und Problemstellung

Unternehmen beziehen zunehmenden IT-Leistungen aus Cloud-Umgebungen, um die Digitalisierung voranzutreiben, ihre Wirtschaftlichkeit zu steigern und um vor allem auch schneller auf Marktbedarfe reagieren zu können. IT-Organisationen transformieren zu agilen Zusammenarbeitsmodellen, um in immer kürzeren Zyklen IT-Systeme und Anwendungen anpassen und bereitstellen zu können. Dabei werden Anwendungen in immer kleinere Einzelteile gegliedert, sogenannte Microservices, die sich hoch dynamischer und skalierbarer Cloud-Infrastruktur, wie virtualisierten Umgebungen, Containern oder sogar Serverless Modellen bedienen.

Nicht direkt Nutzen bringende IT-Abläufe in der Entwicklung werden stark automatisiert, wie z.B. das Testmanagement, sowie die Build-, Integration- und Deploymentprozesse. Dies gilt auch für das Infrastrukturmanagement mittels Methoden wie Infrastructure-as-Code (IaC). Die Nutzung einfach wiederverwendbarer Bausteine, wie Docker-Containern mit vorinstallierter Software, nimmt deutlich zu, um die Effizienz und den Durchsatz in der Anwendungsentwicklung zu maximieren. Diese IaC- oder Container-Vorlagen können aus öffentlichen Registries heruntergeladen werden. Bekannt sind Docker Hub, Terraform, oder AWS CloudFormation. Jedoch können Vorlagen dort von jedem hinterlegt werden. Auch ohne einheitliche Sicherheitsprüfung, oder sogar mit maliziöser Motivation.

Dieser Wandel schafft nicht nur Vorteile, sondern stellt insbesondere auch die Informationssicherheit vor Herausforderungen und kann zu einem erhöhten Risikopotential für Unternehmen führen. Während die Informationssicherheit die Überwachung produktiver Systemlandschaften und der Netzwerke zunehmend automatisiert, werden Auditierungen und Prüfungen der IT-Systeme auf deren Sicherheit oftmals noch mittels zeit- und kostenaufwendiger manueller Abläufe ausgeführt. Dazu erarbeitete Sicherheitsrichtlinien sind oftmals ausführlich dokumentiert, jedoch nicht maschinell lesbar und benötigten Sicherheitsfachkompetenz. Dies führt offensichtlich zu Zielkonflikten.

Aktuelle Cloud-Sicherheitsanalysen zeigen Sicherheitsprobleme die hieraus resultieren können. Palo Alto Networks‘ Threat Intelligence Team Unit42 berichtet jüngst, dass

- 63% der Terraform Vorlagen eine oder mehrere unsichere Konfigurationen aufweisen
- 49% der Vorlagen sogar mindestens eine kritische oder sehr unsichere Konfiguration

Auch bei AWS CloudFormation Vorlagen auf Github wurden derartige Konfigurationsfehler festgestellt. Hier waren es 42% der 1 Million untersuchten Vorlagen mit konfigurationsbedingten Sicherheitsmängeln [1]. Auch die National Security Agency (NSA) kommt 2020 in ihrem Informationsbericht zur Cloud-Sicherheit zur Einschätzung, dass kundenseitige Fehlkonfigurationen in Cloud-Infrastrukturen die höchste Prävalenz aufweisen, bei gleichzeitig geringster benötigter Expertise der Angreifer, um diese Schwachstellen auszunutzen [2]. Dies stellt eine gefährliche Kombination dar, da der wirtschaftliche Aufwand für den Angreifer dadurch geringer ausfällt. Die Analyse von IBM Security kommt ebenfalls 2020 zu dem Ergebnis, dass Cloud-Fehlkonfigurationen mit 19% der häufigste initiale Angriffsvektor sind [3]. Der Bericht beziffert die durchschnittlichen Kosten je Sicherheitsvorfall durch Cloud-Fehlkonfigurationen mit $4.41 Mio USD. Hillary Baron et al. stellen in ihrer Studie für die Cloud Security Alliance (CSA) fest, dass 44% der befragten Organisationen eine Woche oder mehr benötigen, um Sicherheitsvorfälle aus Cloud-Fehlkonfigurationen zu entdecken [4]. Die Zeit bis zur Behebung ist darin noch nicht eingerechnet.

1.2 Motivation

Es stellt sich die Frage, welche Ansätze und Methoden zur Identifikation und Vermeidung dieser sicherheitsrelevanten Cloud-Infrastruktur Konfigurationsfehler hilfreich erscheinen, um die Informationssicherheit der Unternehmen zu verbessern und auch regulatorischen bzw. Compliance-Anforderungen (wieder) zu genügen. Es ist erkennbar, dass klassische Ansätze, wie beispielsweise die Definition von Sicherheitsvorgaben mit manueller Sicherheitsprüfungen, dem Penetrationtesting nach Produktivsetzung oder auch personellen Maßnahmen, allein nicht ausreichend sein werden. Diese können einerseits zu spät greifen, anderseits kollidieren diese tendenziell mit

der Dynamik, Änderungsgeschwindigkeit und Agilität der anderen IT-Prozesse. Personelle Engpässe verstärken dies.

Drei Kriterien erscheinen wesentlich für einen zielführenden Ansatz:

1. Automatisierung der Detektion der unerwünschten Konfigurationen und idealerweise auch der Maßnahmen zur Behebung der Sicherheitsmängel
2. Kontinuierlicher Einsatz zur Überwachung, idealerweise mit minimaler Störwirkung auf IT-Prozesse oder laufende IT-Systeme beziehungsweise die Netzwerke
3. Frühzeitige Erkennung, idealerweise schon vor Produktivsetzung

In der aktuellen Literatur beziehungsweise in Publikationen von Sicherheitsunternehmen lässt sich verstärkt ein methodischer Ansatz, der mittels eines Werkzeugs automatisiert werden soll, erkennen. Gartner Inc. hat dazu den Begriff Cloud Security Posture Management (CSPM) geprägt.

1.3 Forschungsfragen

Die seitens dem Marktforschungsunternehmen Gartner Inc. gesetzten Schwerpunkte zu CSPM, siehe hierzu die Definition in Kapitel 2.2, spannen einen interessanten Rahmen in diesem Spannungsfeld auf, werfen aber auch Fragen auf. Im Rahmen dieser Thesis soll eine Auswahl hierzu relevanter Forschungsfragen im Fokus stehen:

- Welche üblichen Frameworks oder bewährten Sicherheitsverfahren, international meist als *best practises* bezeichnet, zur Verbesserung der Informationssicherheit von Cloud-Konfigurationen im Liefermodell Infrastructure as a Service (IaaS) sind derzeit identifizierbar?
- Lassen sich diese, oder Teile davon, für eine Cloud-Sicherheitsauditierung sinnvoll automatisieren?
- Welchen Beitrag zur Konformität anerkannter Normen der Informationssicherheit kann ein solches Verfahren perspektivisch bieten?
- Wie kann eine Automatisierung von Sicherheitsrichtlinien unter Nutzung des APIs oder anderer technischer Mittel der Cloud-Plattformen prototypisch, anhand einer ausgewählten Cloud-Umgebung, technisch umgesetzt werden?
- Welche Grenzen dieser Methodik und der Technologie sind erkennbar? Welche Risiken sind erkennbar?

1.4 Aufbau des Dokuments

Die Thesis ist in sieben Kapitel untergliedert, sowie der abschließenden Zusammenfassung in Kapitel 8.

Das *Kapitel 2* dient der Erarbeitung der notwendigen Grundlagen im Kontext der Cloud-Dienstmodelle, sowie der Zuständigkeiten und umfasst eine Cloud Security Posture Management (CSPM) Begriffsdefinition.

In Bild 1 ist die Abfolge, die wesentlichen Inhalte und Ergebnisse, sowie das logische Zusammenspiel der nun folgenden Kapitel veranschaulicht.

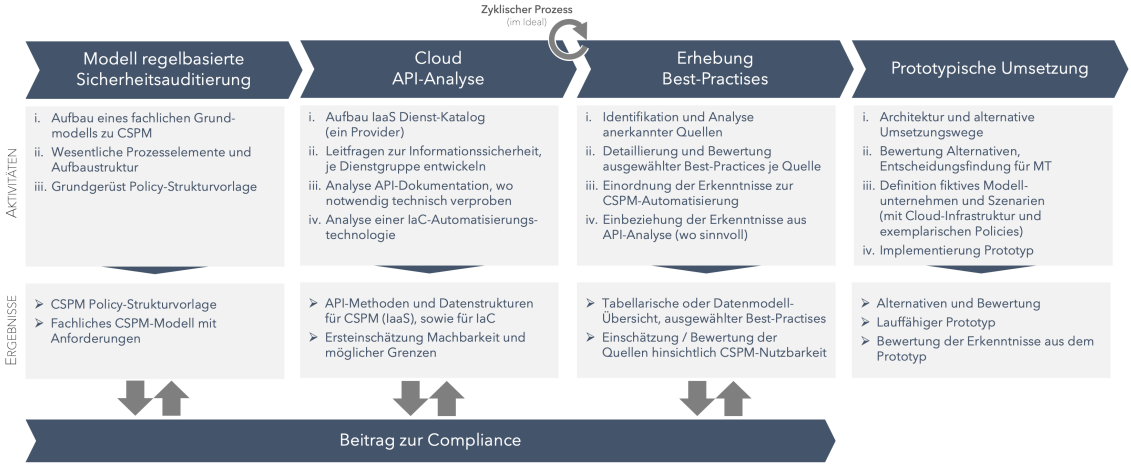

Bild 1: Methodik und Vorgehensmodell der Masterthesis

In *Kapitel 3* wird ein Modell zur automatisierten Sicherheitsauditierung von Cloud-Infrastrukturen erarbeitet. Das Modell umfasst eine statische Sicht der interagierenden Parteien und technischen Komponenten, sowie eine prozessuale Modelldarstellung. Das erarbeitete Prozessmodell wird dabei einem aktuellen Sicherheitsrahmenwerk gegenübergestellt, um ein mögliches Zusammenwirken zwischen dem Prozessstandard und dem entwickelten Prozessmodell erkennen zu können. Die wesentlichen funktionalen Anforderungen an ein System zur automatisierten Sicherheitsauditierung werden zusammengestellt.

Das *Kapitel 4* analysiert die Programmierschnittstelle eines Cloud-Anbieters anhand entwickelter Sicherheitsleitfragen, um die Möglichkeiten und eventueller Grenzen der automatisierten Sicherheitsauditierung zu ermitteln. Dazu musste zunächst ein Dienstkatalog erarbeitet werden, um eine klare Abgrenzung auf die IaaS-relevanten

Dienste geben zu können. Die Analyse erfolgt dabei anhand gebildeter IaaS - Dienstgruppen und fasst die Erkenntnisse der sicherheitsrelevanten API-Operationen, sowie der identifizierten Schlüssel und Werte tabellarisch zusammen. Dabei werden die Erkenntnisse stets den Sicherheitsleitfragen klar zugeordnet.

In *Kapitel 5* erfolgt eine Marktanalyse, welche bewährten Cloud-Sicherheitsrichtlinien ausgewählter Sicherheitsinstitutionen prinzipiell verfügbar sind und ob diese für eine automatisierte Sicherheitsauditierung zielführend eingesetzt werden könnten. Die einzelnen Quelle werden dabei detaillierter untersucht, die Erkenntnisse jeweils zusammengefasst und es erfolgt eine Einschätzung. Die ausgewählten Quellen werden abschließend anhand eines entwickelten Bewertungsschemas beurteilt und bewertet.

In *Kapitel 6* wird auf Basis der gewonnen Erkenntnisse aus der Analyse der Programmierschnittstelle, im Sinne derzeit gegebener technischer Möglichkeiten und Grenzen, sowie der Ergebnisse aus den Best-Practises, als ein Katalog möglicher Sicherheitsrichtlinien, eine prototypische Umsetzung auf Basis einer ausgewählten technischen Variante beispielhaft implementiert. Die Umsetzung prüft dazu ausgewählte Sicherheitsrichtlinien gegen ein erstelltes Szenario einer Cloud-Infrastruktur-Konfiguration, die eingebaute Sicherheitsmängel umfasst. Diese sind automatisiert zu erkennen und dem Aufrufer in Form eines Prüfprotokolls auszuweisen.

Welchen Beitrag zur Verbesserung der Sicherheitskonformität der CSPM-Ansatz liefern könnte, wird in *Kapitel 7* diskutiert. Es werden dazu ausgewählte Sicherheitsnormen und deren Kontrollen den Fähigkeiten der automatisierten Erkennung von Sicherheitsmängeln in Cloud-Konfigurationen gegenübergestellt. Es erfolgt eine Bewertung und Einschätzung.

2 Grundlagen

2.1 Cloud Servicemodelle

Cloud-basierte Dienste teilen grundlegende Charakteristiken, wie die Poolbildung von Ressourcen, einem leistungsfähigen Netzwerkzugang, kurzfristig und skalierbare, sogenannte elastische, Bereitstellung von Kapazitäten, als auch einer nutzungsbasierten Abrechnung und einer an der Selbstbedienung orientierten Bereitstellung. Die Angebotsvielfalt an cloud-gestützten Diensten steigt stetig, diese lassen sich jedoch weitgehend in drei wesentliche Cloud Servicemodelle gliedern [5, 6].

Infrastructure as a Service

Im Servicemodell Infrastructure as a Service (IaaS) werden IT-Ressourcen wie z. B. Rechenleistung, Datenspeicher oder Netze als Dienste angeboten. Der Cloud-Kunde erwirbt diese Leistungen über ein Mietmodell. Die Dienste werden dabei hoch standardisiert angeboten und durch den Kunden über einheitliche Schnittstellen konfiguriert. Der Cloud-Kunde baut darauf seine Dienste in Form von Anwendungen auf und kann diese intern oder extern nutzen. Dazu notwendige Betriebssysteme werden vom Cloud-Anbieter, bzw. dessen Partnernetzwerk, angeboten. Die Konfiguration, die Einstellungen und die Installation weiterer Software nimmt der Kunde vor, sowie auch deren Wartung und Betrieb auf der Cloud-Infrastruktur des CSP. Wird mehr Leistung, beispielsweise in Form von Arbeitsspeicher oder Speicherplatz benötigt, kann dies innerhalb kurzer Zeit automatisiert bereitgestellt werden.

Platform as a Service

Ein Platform as a Service (PaaS)-Provider stellt sowohl die Infrastruktur (Netztechnologie, Rechenleistung, Speicher, Speicher- und Netzwerksicherheit usw.) bereit, als auch auf der Plattform standardisierte Schnittstellen zu Diensten, die von

Anwendungen des Kunden genutzt werden. Die darunterliegenden Betriebssysteme und für die Plattform-Dienste notwendigen Anwendungen, wie beispielsweise Datenbanken oder Identitätsdienste, werden vom PaaS-Provider bereitgestellt und betrieben. Der Cloud-Kunde hat darauf keinen Einfluss. Gegenüber IaaS werden Dienste wie Middleware, Entwicklungswerkzeuge, Reporting/BI-Dienste, oder Datenbankverwaltungssysteme oder auch Containerorchestratoren angeboten. Eine PaaS-Plattform dient dem Kunden als eine standardisierte und verwaltete Grundlage zur Entwicklung und dem Betrieb eigener Anwendungen und Apps.

Software as a Service

Sämtliche Angebote von Anwendungen die auf Basis von Cloud-Computing angeboten werden, sind dem Servicemodell Software as a Service (SaaS) zuzuordnen. Die Vielfalt der Angebote auf dem Markt ist hier sicherlich am breitesten. Software as a Service (SaaS) Lösungen reichen von populären Kommunikations- und Kollaborationsdiensten wie Microsoft Teams oder Zoom, über Kundenmanagement- und Vertriebslösungen wie Salesforce oder Successfactor, cloud-basierter Textverarbeitung wie Microsoft Office 365 oder Google Docs. Hier nutzt der Cloud-Kunde die Anwendung und hat keinen Einfluss auf die darunter liegenden Ebenen, wie dem Betriebssystem, den Datenbanken oder dem Datensystemen. Oftmals bieten SaaS-Lösungen auch die Integration mit anderen Diensten, die auch in der Hoheit des Kunden sein können. Beispielsweise einem zentralen Identitätsmanagement.

Function as a Service

Der technische Fortschritt bieten immer wieder neue Ansätze und Lösung, die auch zu neuen Begriffsbildungen führen. Ein bekanntes, oft als weitere Kategorie genanntes, Servicemodell wird als Function as a service (FaaS) bezeichnet. Im Zentrum steht hier der Gedanke, dass der Kunde eine Umgebung vom CSP anmieten kann, auf der er Dienste entwickelt, ohne sich um die Entwicklungs- und Betriebsumgebung, dem Betriebssystem oder dessen Infrastruktur zu kümmern. Dieses Modell wird oft im Zusammenhang mit Architekturmuster wie Microservice-Applikationen, oder dem serverless Architekturparadigmas verwendet. Dieses Servicemodell hat erkennbare Ähnlichkeiten zu PaaS. Als ein Unterscheidungsmerkmal könnte genutzt werden, dass es bei einem Function as a Service (FaaS)-Modell aus Sicht des Kunden keine 'Server' mehr gibt. Die Instanziierung notwendiger Diensten, als Ausführungsumgebung für die Dienste und Anwendungen des Kunden, werden von der FaaS-Plattform

selbständig übernommen. Eine FaaS-Umgebung hat keine permanent laufende virtuelle Instanz, oder Containerumgebung, für den Kunden. Diese werden nur bei Bedarf bereitgestellt und in einem pay-per-use Modell abgerechnet.

Bereitstellungsmodelle Public- und Private-Cloud

Neben den Cloud Servicemodellen definiert das National Institute of Standards and Technology (NIST) vier Cloud-Bereitstellungsmodelle (Deployment Model) [5]. Die Bereitstellung wird unterschieden in Private Cloud, Public Cloud, Community Cloud und Hybrid Cloud. Das bekannteste Modell hierbei ist sicherlich die *Public Cloud*, bei der die Cloud-Infrastruktur öffentliche und uneingeschränkt für alle potentiellen Nutzer und Kunden angeboten wird. Die Cloud-Umgebung ist mandantenfähig und isoliert die einzelnen Kunden und deren Daten. Als CSP können Unternehmen, öffentliche Organisationen und Behörden, akademische Institutionen, oder auch gemeinnützige Organisationen auftreten. Eine *Private Cloud* Bereitstellung ist hingegen exklusiv für eine Organisation, die in mehrere abgegrenzte Konsumenten gegliedert sein kann. Beispielsweise für einzelne Geschäftseinheiten. Eine *Community Cloud* ist ähnlich einer Private Cloud und für einen geschlossenen Kreis von Konsumenten ausgelegt, die einer Gruppe oder einem Verbund von gleichartigen Interessenten oder Eignern angehören. Eine *Hybrid-Cloud* stellt eine Kombination der vorherigen Modelle dar.

Aus Sicht des in der Thesis bearbeiteten Ansatzes des CSPM sollte das Bereitstellungsmodell keinen wesentlichen Unterschied darstellen. Der CSPM-Ansatz sollte sowohl für Private- als auch für Public-Cloud Modelle einsetzbar sein, solange die Cloud-Plattform über ausreichende Zugriffsmöglichkeiten über Programmierschnittstellen zur automatisierten Auditierung verfügt. Im Rahmen der Thesis wird aber der Einsatz einer Public-Cloud Plattform im Fokus stehen.

2.2 Definition Cloud Security Posture Management

Der Begriff Cloud Security Posture Management (CSPM) geht zurück auf einen Bericht des Analysten Neil MacDonald, der für das Technologie- und Marktforschungsunternehmen Gartner Inc. tätig ist [7].

Gartner's Definition des Cloud Security Posture Management Ansatzes bzw. dessen Leistungsumfang beschreibt MacDonald wie folgt:

> *CSPM offerings continuously manage cloud risk through the prevention, detection, response and prediction of where excessive cloud infrastructure risk resides based on common frameworks, regulatory requirements and enterprise policies. The core of CSPM offerings proactively and reactively discover and assess risk/trust of cloud services configuration (such as network and storage configuration), and security settings (such as account privileges and encryption). Ideally, if a setting is noncompliant or a configuration represents excessive risk, the CSPM offering can take automated action to adapt, including remediation.*

Hinsichtlich der drei wesentlichen Cloud-Servicemodelle (IaaS, PaaS und SaaS) beschreibt MacDonald konkretisierend weiter:

> *The primary purpose of CSPM offerings is to continuously identify and remediate cloud infrastructure risks consistently across all of the cloud IaaS and PaaS platforms in use by an organization.*

Mit Bezug zu PaaS Diensten nennt Gartner explizit *serverless* Dienste, wie Lambda von AWS bzw. Functions bei Microsoft Azure.

Erläuternd zur Definition ergänzt Gartner einige Anwendungsfälle und Mehrwerte, die die Analysten für CSPM erwarten (Auszug):

- *Continuous discovery and identification of cloud workloads and services*
- *Policy visibility and consistent enforcement across multiple cloud providers*
- *Alerting on risky new deployments or changes to the cloud environment, hosts or services*
- *Riskassessment versus frameworks and external standards (such as the International Organization for Standardization and National Institute of Standards and Technology).*

David Puzas von Crowdstrike Inc., einem renomierten US-amerikanischen Unternehmen für Informationssicherheit, definiert CSPM wie folgt [8]:

> *Cloud security posture management (CSPM) automates the identification and remediation of risks across cloud infrastructures, including Infrastructure as a Service (IaaS), Software as a Service (Saas), and Platform as a Service (PaaS). CSPM is used for risk visualization and assessment, incident response, compliance monitoring, and DevOps integration, and can uniformly apply best practices for cloud security to hybrid, multi-cloud, and container environments.*

Bei Puzas's Definition werden die wesentlichen Elemente des Ansatzes von Gartner aufgegriffen. Die etwas erweiterte Definition wirkt vertrieblich ausgerichtet auf CrowdStrike's Sicherheitsprodukte.

In Veröffentlichungen aus dem Bereich der Forschung finden sich nur wenige Artikel mit direktem Bezug zu CSPM. Bolannavar übernimmt in seiner Publikation im Wissenschaftsjournal IJSCSEIT weitgehend die wesentlichen Elemente aus der Definition von Gartner [9].

> *CSPM offerings continuously manage cloud risk through the prevention, detection, response and prediction of where excessive cloud infrastructure risk resides based on common frameworks, regulatory requirements and enterprise policies. The core of CSPM offerings proactively and reactively discover and assess risk/trust of cloud services configuration (such as network and storage configuration), and security settings (such as account privileges and encryption). Ideally, if a setting is noncompliant or a configuration represents excessive risk, the CSPM offering can take automated action to adapt, including remediation.*

Der Begriff Cloud Security Posture Management (CSPM) wird weiterhin von Softwareherstellern aus dem Bereich der Informationationssicherheit verwendet. Diese fassen teilweise die Definition etwas weiter und beziehen die Möglichkeit ein eine API-basierte Analyse auch auf SaaS-Anwendungen anzuwenden, wenn diese ausreichende Zugriffsmöglichkeiten bereitstellen. Die Prinzipien bleiben hierbei jedoch bestehen, wodurch die ursprüngliche Definition von Gartner als allgemeingültig anerkannt werden könnte.

Im Rahmen dieser Thesis wird die Definition von Gartner verwendet und der Fokus auf das Cloud-Servicemodell Infrastructure as a Service (IaaS) gelegt.

2.3 Verantwortlichkeiten in der Cloud

Werden Rechenleistungen und Daten in eine Cloud-Umgebung ausgelagert und dort betrieben, stellt sich die Frage der Verantwortlichkeiten für die Informationssicherheit. Während bei einem traditionellen Betrieb eines Rechenzentrums in den unternehmenseigenen Gebäuden die Verantwortlichkeit nahezu vollständig bei dem Unternehmen selbst liegt, sind in einer Cloud-Umgebung mehrere Parteien verantwortlich. Die Verantwortlichkeit verlagert sich auch nicht vollständig auf den Cloud-Anbieter.

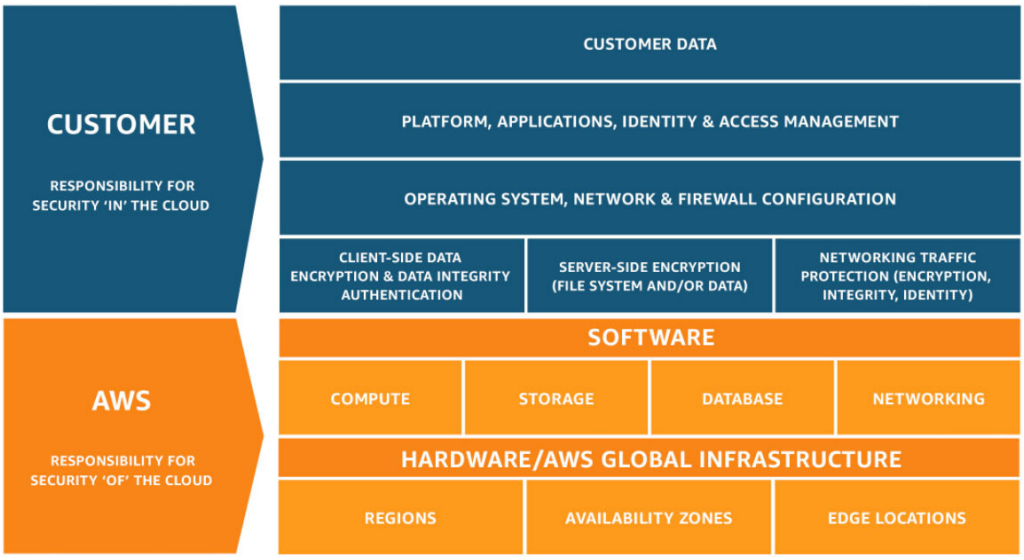

Bild 2: AWS Modell der geteilten Verantwortung (Bildquelle: AWS [10])

Üblicherweise erstellen Cloud Service Provider (CSP) ein sogenanntes *Modell der geteilten Verantwortung* für deren Cloud-Dienste, um die Verantwortlichkeiten des Kunden und des Anbieters zu bestimmen und nachfolgend mit vertraglichen Vereinbarungen zu regeln. Als ein Beispiel ist in Bild 2 das Modell des Anbieters Amazon Web Services (AWS) dargestellt [10]. Während der Cloud-Anbieter für die Hardware, eine globale Infrastruktur mit Gebäuden und Rechenzentren, die Bereitstellung von Rechen-, Speicher- und Netzwerkleistung verantwortlich ist, ist der Kunde für seine Daten, Plattformen und Anwendungen die er auf der Infrastruktur des CSP installiert verantwortlich. Auch liegt in der Verantwortung des Kunden der Betrieb, die Konfiguration und die Wartung der Betriebssysteme, seiner angemieteten virtualisierten Netzwerke und der Einsatz von Sicherheitskomponenten wie Firewall- oder IDS/IPS-Systeme. Maßnahmen zur Verbesserung der Informationssicherheit, bei-

spielsweise durch zeitnahe Aktualisierung der Betriebssysteme, einer abgesicherten Netzwerktopologie, dem Identitäts- und Zugriffsmanagement, oder die Verschlüsselung der Daten auf den angemieteten virtualisierten Speichersystemen bzw. im Netzwerkverkehr, sind in der Verantwortung des Kunden.

Diese Regelung der Verantwortlichkeit spielt im Kontext der automatisierten Prüfung und Absicherung von Cloud-Diensten und deren Konfiguration, wie dies mittels Cloud Security Posture Management (CSPM) erfolgen könnte, eine wichtige Rolle. Die Verantwortung für die korrekte und sichere *Konfiguration* der angemieteten Cloud-Dienste liegt, sowohl für funktionale, nicht-funktionalen Aspekte und der Parametrisierung bezüglich der Informationssicherheit der Ressourcen, maßgeblich in der Verantwortung des Kunden selbst.

Die Verantwortlichkeiten zwischen Cloud-Anbieter und -Kunde unterscheiden sich in Abhängigkeit des genutzten Cloud-Servicemodells (IaaS, PaaS, oder SaaS). Die Verantwortung des Cloud-Anbieters ist bei SaaS deutlich umfangreicher, da hier der Kunde weitaus weniger Einfluß nehmen kann auf Dienste wie beispielsweise das Betriebssystem, die Speicher und deren Konfiguration. Das Modell des Anbieters Microsoft geht hierauf detaillierter ein [11]. Insbesondere im Servicemodell PaaS kann es in verschiedenen Ebenen sogar zu einer gemeinsamen Verantwortung kommen, bzw. sind hier sehr genaue Regelungen zwischen den Parteien zu treffen.

2.4 Standards und Rahmenwerke der Informationssicherheit

Der strukturierte Aufbau eines Managementsystems für die Informationssicherheit (ISMS), sowohl bei Behörden als auch bei Unternehmen, wird meist auf Basis eines ausgewählten und spezialisierten Rahmenwerks für die Informationssicherheit ausgeführt. Diese anerkannten Rahmenwerke oder Standards der Informationssicherheit können auch für eine nachfolgende Überprüfung, Testierung oder Zertifzierung der Organisation, der etablierten Verfahren und Prozesse, sowie der eingesetzten Technik dienen. Eine der international bekanntesten Normen für die Informationssicherheit ist die kostenpflichtige ISO/IEC 27000 Reihe, sowie im deutschsprachigen Raum der IT-Grundschutz mit dem IT-Grundschutz-Kompendium des Bundesamt für Sicherheit in der Informationstechnik (BSI) [12, 13, 14]. Im internationalen Kontext sind die Standards des National Institute of Standards and Technology (NIST), insbesondere das Cybersecurity Framework (CSF) und die Spezial-Publikation 800-53, sowie das Rahmenwerk 'Critical Security Controls' des Center for Internet Security (CIS) anerkannt und in werden in der Praxis genutzt [15, 16, 17].

Diese Standards stellen ein konsolidiertes und gepflegtes Wissen zur Verbesserung der Informationssicherheit dar, das einerseits eine Hilfestellung bei der Implementierung eines Information Security Management System (ISMS) bietet, sowie auch als Kontrollfunktion dienen kann, um den erreichten Sicherheitsstand oder Reifegrad eines Unternehmens zu prüfen und damit auch die Konformität zu den Anforderungen der Sicherheitsstandards strukturiert zu verifizieren. Diese Anforderungen zur Verbesserung der Informationssicherheit werden meist als Kontrollen (englisch Controls) bezeichnet und sind in Kategorien oder Gruppen zusammengefasst.

Das IT-Grundschutzkompendium des Bundesamt für Sicherheit in der Informationstechnik (BSI) ist beispielsweise in zehn sogenannte Prozess- und System-Bautein(gruppen) gegliedert, die wiederum Bausteine enthalten (siehe Bild 3). Die in Summe 68 Bausteine gruppieren fachlich zusammenhängende Anforderungen. So beinhaltet die Bausteingruppe *NET: Netze und Kommunikation* den Baustein *NET.1.1 Netzarchitektur und -design* mit insgesamt einzelnen 36 Anforderungen zur Verbesserung der Informationssicherheit. Eine Anforderung in diesem Baustein fordert die Segmentierung des Gesamtnetzes in einzelne Netzbereiche *NET.1.1.A4 Netztrennung in Zonen (B)*, mindestens in drei Zonen (intern, demilitarisiert (DMZ) und Außenanbindung), sowie beispielsweise eine zweistufige Firewallarchitektur zwischen vertrauenswürdigen und nicht vertrauenswürdigen Netzen u.a. die mittels Paketfiltern abzusichern. Das BSI differenziert die Anforderungen weiterhin nach Basis- und Standardanforderungen, die gemeinsam den Stand der Technik abbilden und umgesetzt werden sollten. Die Basisanforderungen sind hingegen ein muss.

IT-Grundschutz-Kompendium (Edition 2023)

Prozess-Bausteine	System-Bausteine
ISMS | Sicherheitsmanagement	APP | Anwendungen
ORP | Organisation und Personal	SYS | IT-Systeme / Server
CON | Konzepte und Vorgehensweisen	IND | Industrielle IT
OPS | Betrieb	NET | Netze und Kommunikation
DER | Detektion und Reaktion	INF | Infrastruktur

Bild 3: Bausteingruppen des BSI IT-Grundschutz Kompendium

Der Standard von NIST, SP800-53 *Security and Privacy Controls for Information Systems and Organizations*, verfolgt eine ähnliche Struktur und Ansatz. Die Autoren von NIST gruppieren die Kontrollen in 20 sogenannte *Security and Privacy Control Families*, wie beispielsweise die 'Familie' *SC - System and Communications Protection*. Die Kontrollen entsprechen den Anforderungen aus dem BSI IT-Grundschutz

Kompendium. Die Kontrollen sind durchnummeriert und strukturell gleichartig dokumentiert. Die Kontrollen, wie beispielsweise *SC-5 Denial-of-Service protection*, beinhalten meist eine kompakte Anzahl an Kernanforderung und zusätzliche Anforderungen, sogenannte *enhancements*. Eine generelle Klassifizierung nach MUSS und SOLL ist nicht erkennbar.

Das Standardwerk *Critical Security Controls* (Version 8.0) des Center for Internet Security (CIS) umfasst 18 Kontrollen, die jeweils circa 5-20 sogenannte Safeguards beinhalten. Jede Kontrolle ist mit einer Bedarfsbeschreibung und einer relativ konkreten Umsetzungsanweisung dokumentiert. Auch diese Empfehlungen sind meist nicht an eine konkrete Technologie gebunden, sondern stellen allgemeingültig Sicherheitsvorgaben dar. Die Safeguards sind jeweils einem Titel klar benannt und ergänzend durch 2-3 Sätze definiert. Eine Besonderheit dieses Standards ist, dass die Safeguards den sogenannten *Five Functions* des NIST Cybersecurity Framework (CSF) zugeordnet sind [18, 19]. Das CIS gibt auch eine Empfehlung zur Priorisierung der Safeguards. CIS nutzt dazu (seit Version 7.1) drei Stufen, die sogenannte *Controls Implementation Groups* (IG). Die erste Gruppe IG1 stellt die Basis dar und wird als *essential cyber hygiene* tituliert. Die weiteren Stufen IG2 und IG3 bauen darauf auf und bietetn zusätzlichen Schutz. Diese Empfehlung wird durch einen offenen Community-Prozess bestimmt, somit durchaus als sehr praxisnahe zu werten.

Wie bereits erwähnt diesen diese Sicherheitsstandards als Hilfestellung zur Verbesserung der Informationssicherheit durch Konzeption und Umsetzung von Sicherheitsempfehlungen, als auch zur Prüfung der Konformität und Abdeckung des jeweiligen Standards. Diese Standards könnten damit auch hilfreich sein eine Sicherheitsmethode oder -technik, wie Cloud Security Posture Management (CSPM), einzuordnen und den potentiellen Mehrwert dieser Methode aufzuzeigen. Andererseits könnten die Standards auch dienlich sein, Sicherheitsregeln zu konzipieren und später zu automatisieren.

Interessant ist auch, dass die logischen Beziehungen zwischen den Kontrollgruppen und Kontrollen der Rahmenwerken über sogenannte Zuordnungstabellen (englisch mapping) hergestellt werden könnten [20, 15, 21, 22, 23].

3 Fachkonzept Cloud Security Posture Management

3.1 Statisches Modell Cloud Security Posture Management

Auf Basis der in Kapitel 2.2 erfolgten Begriffsdefinition von Cloud Security Posture Management (CSPM) ist zunächst ein statisches Modell zu entwickeln, das die wesentlichen Akteure, die Interaktionen und die Zusammenhänge zwischen den Akteuren und den Cloud-Ressourcen strukturiert.

Die schematische Darstellung in Bild 4 veranschaulicht ein, für diese Thesis erstelltes, statisches Modell für einen CSPM-Lösungsansatz. Zunächst sind die Beteiligten und Ressourcen in die Gruppen Cloud-Konsument und Cloud Service Provider (CSP) zu gliedern. Die Akteure *DevOps Teams/Cloud Admin* entsprechen den Entwicklungs- und Betriebsteams eines Unternehmens. Diese erstellen, konfigurieren, verwalten und betreiben die Cloud-Ressourcen, wie diese aus fachlicher Sicht für die Anwendungen und deren Daten notwendig sind. Dazu nutzen sie die Werkzeuge des Cloud-Anbieters, wie beispielsweise grafische Web-Managementoberflächen, oder auch von Drittanbietern. Diese Werkzeuge erstellen, ändern und löschen Cloud-Ressourcen.

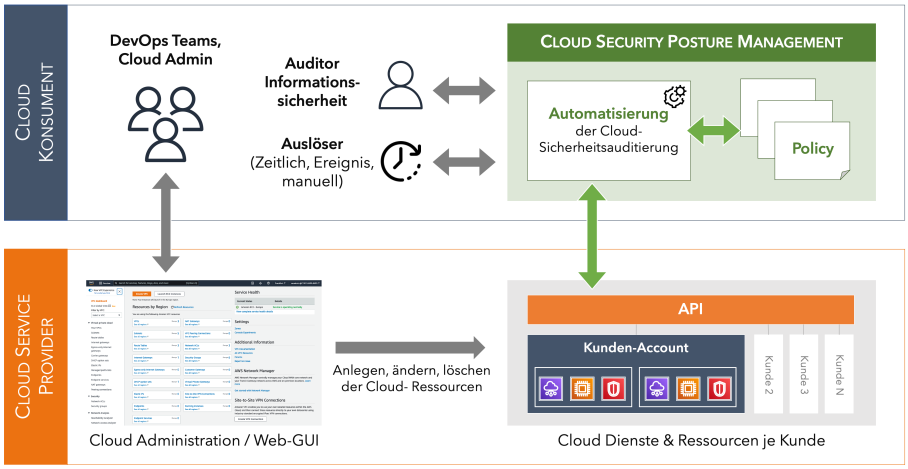

Bild 4: Schematische Darstellung CSPM

Der CSP stellt Dienste und Ressourcen bereit, verwaltet diese je Kunde (englisch tenant) in isolierten Daten- und Rechenbereichen und stellt ein Application Programming Interface (API) zur Verwaltung dieser Ressourcen bereit. Den Leistungsumfang, sowie das Release-Management, dieses API bestimmt der CSP.

Das Cloud Security Posture Management (CSPM)-Modul, grün dargestellt in Bild 4, interagiert über diese standardisierte Schnittstelle mit der Cloud-Plattform. Das CSPM-Modul verfügt über automatisierte Sicherheitsrichtlinien, die in einem maschinenlesbaren Format vorliegen. Die CSPM-Automatisierungskomponente dient als zentrale Ablaufumgebung, die das API nutzt um die Cloud-Ressourcen und deren Konfiguration auszulesen, sowie die erhobenen Informationen den maschinenlesbaren Sicherheitsrichtlinien zuführt und diese ausführt. Bei diesem SOLL-IST-Vergleich stellen die Konfiguration das IST dar und die Vorgaben der Sicherheitsrichtlinien das SOLL. Sicherheitsmängel stellen die Abweichungen zwischen dem SOLL und dem IST dar und werden detektiert. Die Sicherheitsrichtlinie stellt idealerweise auch Hinweise zur Behebung der Sicherheitsmängel bereit, oder automatisiert sogar die Korrektur.

Der Akteur *Auditor Informationssicherheit* bedient das Cloud Security Posture Management (CSPM)-Modul, erstellt und verwaltet maschinenlesbare Sicherheitsrichtlinien. Der Auditor kann auch Sicherheitsanalysen automatisiert ausführen. Die Ausführung der CSPM-Analysen kann durch verschiedene *Auslöser* erfolgen. Analysen könnten manuell, in konfigurierbaren zeitlichen Intervallen oder auch bei Eintreten eines Ereignisses stattfinden. Ein Ereignis kann beispielsweise der Wunsch einer Konfigurationsänderung sein, die durch einen Schritt in einem Software-Entwicklungsprozess ausgelöst wird oder auch das Eintreten einer Konfigurationsänderung, die durch einen Änderungszeiger der Cloud-Plattform erkannt und automatisch ausgelöst werden würde.

Das CSPM-Modul liefert dem Akteur *Auditor Informationssicherheit* die Ergebnisse der automatisierten Sicherheitsprüfungen. Dies kann im einfachsten Fall durch Ausgaben auf der Kommandozeile oder in Protokolldateien erfolgen. Die Interaktion kann aber auch direkt zwischen dem Auslöser, beispielsweise einem Entwicklerteam das ein Release prüfen lässt, und dem CSPM-Modul erfolgen.

3.2 Prozessuale Sicht

Ergänzend zu dem statischen Modell ist eine prozessorientierte Sicht des Informationssicherheitsmanagements aufzubauen, wie dieses mit Unterstützung der CSPM-Automatisierung, gestaltet werden kann. Unternehmen und Organisationen entwickeln deren ISMS auf Basis von anerkannten Standard-Rahmenwerken, wie beispielsweise der ISO 27000 Normenserie, dem NIST Cybersecurity Framework oder auch auf Basis der IT-Grundschutz Methodik des BSI.

Im Rahmen dieser Arbeit wurde ein Versuch unternommen die CSPM-Prozessschritte zu gliedern und diese dem Prozessablauf der Standard-Absicherung aus dem BSI IT-Grundschutz gegenüberzustellen, beziehungsweise diese darin zu integrieren. Der Gesamtprozess lässt sich in fünf Teilprozesse aufteilen, die jeweils wieder einzelne Prozessschritte aufgeteilt werden können. Vier der fünf Prozesse lassen sich dem Prozessablauf der Standard-Absicherung zuordnen. Der fünfte Prozess, die *Pflege und Erweiterung des CSPM Policy-Katalogs*, ist spezifisch für vorbereitenden Tätigkeiten der CSPM-Automatisierung. Die fünf Ebenen der CSPM-Teilprozesse und die Gegenüberstellung zur Standard-Absicherung sind in Bild 5 dargestellt. Automatisierbare Prozessschritte sind mit einem Zahnrad-Symbol gekennzeichnet, manuelle Tätigkeiten bleiben ohne Symbol.

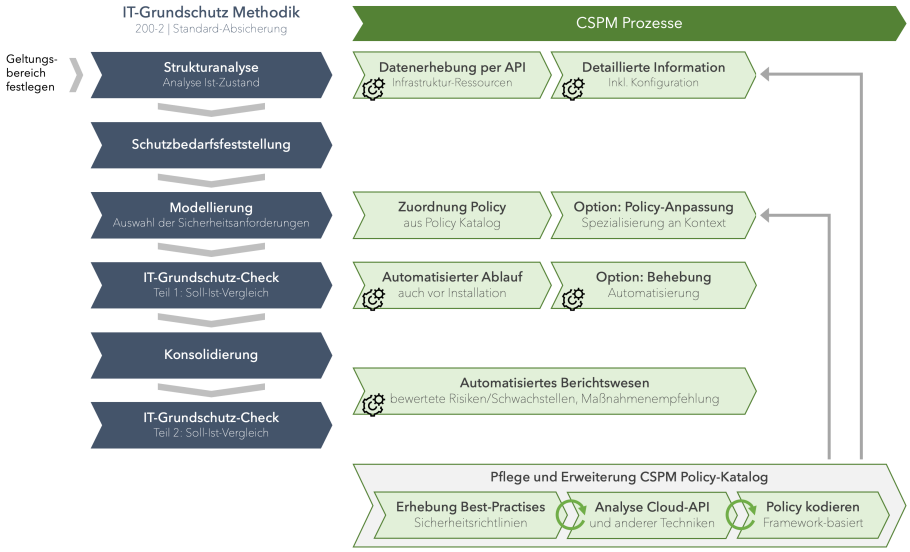

Bild 5: BSI IT-Grundschutz Methodik und schematischer CSPM-Prozess

Die oberste Teilprozessebene dient der Ermittlung des Ist-Zustands der Cloud-Infrastruktur und kann in zwei Prozessschritten erfolgen. Zunächst erfolgt die *Da-*

tenerhebung per API auf der Detaillierungsstufe der Cloud-Ressourcen, die in einem zweiten Schritt um die Details der Ressourcen-Konfiguration automatisiert ergänzt werden kann. Diese Tätigkeiten finden typischerweise im Rahmen der *Strukturanalyse* im IT-Grundschutz statt. Strukturinformationen, wie beispielsweise Gebäude und Räume, können nicht oder nur sehr abstrakt im Falle einer Cloud-Nutzung erhoben werden. Nicht virtualisierte Strukturen können nicht automatisiert erfasst werden.

Die *Schutzbedarfsfeststellung* im IT-Grundschutz ist typischerweise ein manueller Vorgang und findet bei CSPM keine automatisierende Unterstützung.

Die zweite Teilprozessebene unterstützt oder wirkt auf der Ebene der *Modellierung* des IT-Grundschutzes. Im Prozessschritt *Zuordnung Policy* ordnet der Sicherheitsexperte den existierenden oder zukünftigen Cloud-Ressourcen die anzuwendenden Sicherheitsrichtlinien, aus einem bestehenden CSPM-Policy-Katalog, zu. Beispielsweise könnten die Zuordnung der Sicherheitsrichtlinien, die für eine Entwicklungslandschaft einzusetzen sind, abweichend sein von denen für eine Produktionsumgebung. Ebenso können sich anzuwendende Sicherheitsrichtlinien in Abhängigkeit zur geographischen Lage der Daten und Systeme ändern. Optional könnten hier auch *Anpassungen* an die Policy-Vorlage aus dem Katalog erfolgen.

Die dritte Ebene stellt die *automatisierte Auditierung* der Cloud-Ressourcen dar. Diese kann einmalig, wiederkehrend oder zu gewissen Ereignissen erfolgen. Die zuvor konfigurierten Sicherheitsrichtlinien werden automatisiert gegen die bestehenden (oder auch zukünftigen) Cloud-Ressourcen und -Konfigurationen angewendet. Soweit sinnvoll und möglich kann als *Option eine Behebung* oder Behandlung erkannter Sicherheitsmängel erfolgen. Dieser Prozessschritt entspricht dem IT-Grundschutz-Check, dem Soll-Ist-Vergleich zwischen definierten Sicherheitsanforderungen und dem aktuellen Stand in dem betrachteten Verbund.

Ebene vier stellt das *automatisierte Berichtswesen* dar. Die Ergebnisse der automatisierten Auditierung, vor allem die detektierten Sicherheitsmängel, werden mit Bezug zu den vulnerablen Cloud-Ressourcen und deren Konfigurationsfehler berichtet und ausgegeben. Das Berichtswesen umfasst idealerweise auch Maßnahmenempfehlungen zur Behebung der Sicherheitsdefizite. Das Berichtswesen kann als Basis für eine Bewertung von Risiken dienen.

Die fünfte Prozessebene stellt einen kontinuierlichen dreistufigen Prozess dar, der auf Basis der Erkenntnisse ausgewählter Sicherheitsorganisationen bzw. aus selbst gewonnenen Erfahrungen bewährte Sicherheitsrichtlinien zur Absicherung der Cloud-

Konfiguration ableitet. Auf Basis der bewährten Sicherheitsrichtlinien wird die Cloud-Programmierschnittstellen, das API des CSP, untersucht, wie eine Automatisierung der Detektion erfolgen kann. Die API-Analyse versucht die Operationen, sowie die Daten (Schlüssel- und Wertpaare), zu ermitteln und Logikregeln zu erarbeiten. Auf Basis der gewonnenen Analyse-Erkenntnisse werden automatisierte Richtlinien entwickelt und für die weitere Anwendung im CSPM-Prozess in einem Katalog bereitgestellt. Die automatisierten Richtlinien werden dabei idealerweise zu relevanten Kontrollen aus Sicherheitsrahmenwerken zugeordnet und wo sinnvoll möglich hinsichtlich möglicher Risiken klassifiziert.

3.3 Funktionale Anforderungen Cloud Security Posture Management

Für die spätere Umsetzung einer prototypischen CSPM-Lösung sind die wesentlichen Anforderungen zu benennen. Eventuell können diese später nicht gänzlich im Prototyp umgesetzt werden, sollten jedoch für den gewählten Ansatz, mindestens durch eine nachfolgende Erweiterung einfügt oder ergänzt werden können.

3.3.1 Automatisierte Erhebung relevanter Cloud-Ressourcen

Eine automatisierte Erkennung von Schwachstellen in der Cloud-Konfiguration verlangt zunächst die Cloud-Ressourcen und deren Konfiguration automatisiert erheben und erfassen zu können. Dazu hat die CSPM-Lösung die Fähigkeit zu besitzen, die Konfiguration von einem Cloud-Provider automatisiert über eine Programmierschnittstelle strukturiert abfragen und einlesen zu können. Die abgefragten Daten müssen strukturiert weiterverarbeitet werden können, idealerweise in einem technisch standardisierten Dateiformat.

Es ist zu erwarten, dass sowohl das API, die technische Darstellung der Cloud-Ressourcen, sowie deren Konfigurationen sehr spezifisch für den gewählten Cloud Service Provider (CSP) sein werden. Es wird nicht erwartet, dass die Lösung alle CSP-Umgebungen unterstützt.

Die Erhebung der Cloud-Konfiguration sollte eine vollständige Erhebung, als auch ein teilweise Erhebung zulassen. Beispielsweise eingeschränkt auf definierbare Bereiche der Cloud-Konfiguration eines Mandanten, oder, in einem fortgeschrittenen Zustand, nur die aktuellen Änderungen. Dies würde aber Fähigkeiten der Cloud-Umgebung, wie einem Änderungszeiger, voraussetzen.

Die automatisierte Erhebung sollte die Daten entweder nach Bedarf einlesen können oder in einer Vorverarbeitung abrufen und zwischenspeichern, um diese für die Folgeverarbeitung bereitzustellen. Eine Zwischenspeicherung scheint für umfangreiche Cloud-Umgebungen zielführend, um die Laufzeiten und die Belastung des API des CSP in einem vertretbaren Umfang zu halten.

Die Analyse der erhobenen Rohdaten Daten kann in zwei Varianten untergliedert werden. Beiden ist jedoch gleich, dass die erhobenen Rohdaten semantisch interpretiert werden können müssen, sowie die sicherheitsrelevanten Konfigurationselemente fachlich identifiziert und deren Werte im Ist-Zustand erkannt werden. Diese semantische Erkennung bildet für die nachgelagerte Bewertung, ob diese Konfiguration dem erwarteten Soll-Zustand entspricht, eine wesentliche Grundlage.

Analyse bestehender Cloud-Ressourcen

Die über das API erhobenen Cloud-Konfigurationsdaten stellen den Ist-Zustand dar, der bereits in der Cloud produktiv konfiguriert und auch aktiv nutzbar sein kann. Dieser Zustand kann Sicherheitsmängel aufweisen, die durch eine fehlerhafte Konfiguration seitens des Cloud-Kunden entstanden sind und sich bereits aktiv auswirken könnten. Die API-basierte Datenerhebung liefert diese Informationen und gegebenenfalls müssen diese Daten analysiert und für die spätere Bewertung vorverarbeitet werden, wie beispielsweise eine Wandlung aus einem API-Objekt in ein hierarchische Datenformat oder die Interpretation gewisser statischer Werte in fachlich auswertbare Sicherheitsinformationen.

Der funktionale Bedarf lässt sich zum jetzigen Zeitpunkt noch nicht genauer fassen und ergibt sich auf Basis der in Kapitel 4 entstehenden Erkenntnisse und Anforderungen.

Analyse Cloud-Ressourcen vor Änderung der Infrastruktur

Alternativ, im Falle von Konfigurationsvorlagen, wie diese bei Infrastructure as Code (IaC) zum Einsatz kommen, muss die CSPM-Lösung die strukturiert in Dateien abgelegten Cloud-Konfigurationsdaten einlesen und semantisch interpretieren können. Eine API-Erhebung findet in diesem Fall bewusst nicht statt.

Die Konfigurationsdateien können prinzipiell auf Basis verschiedener Standards entstehen und unterschiedliche Formate und Syntax aufweisen. Die Unterstützung des

gängigen JSON-Dateiformats, sowie der Ressourcen-Syntax des aktuellen Marktführers HashiCorp und dessen Produkt Terraform, werden für eine ausgewählte Cloud-Variante und einem Ausschnitt an Cloud-Ressourcen in dieser MT als Anforderung definiert.

Anzumerken ist, dass in dieser Variante eine Analyse vor der eigentlichen Installation der Konfiguration in eine konkrete Cloud-Umgebung stattfinden kann.

3.3.2 Verwaltung von Sicherheitsrichtlinien

Die Sicherheitsrichtlinien beschreiben den Soll-Zustand einer Cloud-Konfiguration, genauer gesagt der Cloud-Ressourcen in einer Cloud-Umgebung. Die Sicherheitsrichtlinien werden in einem zu definierend technischen Format erstellt und gespeichert, sodass diese im Rahmen der Sicherheitsauditierung automatisiert genutzt werden können. Eine CSPM-Lösung sollte die Möglichkeit bieten, diese Sicherheitsrichtlinien zu katalogisieren und zu verwalten, sodass spätere Nutzer diese möglichst einfach auffinden und nutzen können. Im einfachsten Fall kann dazu eine strukturierte Ablage, gemeinsam mit einer einfachen Versionsverwaltung und einer aussagekräftigen Identifikation und Deskription der Sicherheitsregeln über menschlich und maschinell lesbaren Metadaten, genügen.

In einer fortgeschrittenen Version könnte eine webbasierte grafische Suchoberfläche, die auch eine Selektion nach Cloud-Ressourcenarten oder nach fachlichen Merkmalen wie der potentiellen Sicherheitskritikalität ermöglicht einen Mehrwert darstellen. Eine Integration in bestehende Sicherheits- oder Entwicklungswerkzeuge wäre prinzipiell auch denkbar und wahrscheinlich erstrebenswert, geht aber über den Rahmen dieser Thesis hinaus.

Als fachliche Vorgabe wurde die Grundstruktur einer Sicherheitsrichtlinien erarbeitet. In Bild 6 ist die generelle Struktur dieser Vorlage einer Sicherheitsrichtlinie, sowie ein Beispiel dargestellt.

Die genauen fachlichen und technischen Inhalte einer solchen maschinell lesbaren Sicherheitsrichtlinien werden im Nachgang der Erkenntnisse aus der Analyse des API in Kapitel 4, den Ergebnissen aus der Marktanalyse zu den bewährten Methoden (englisch Best-Practises) aus Kapitel 5 und der gewählten Umsetzungsvariante in Kapitel 6 konkretisiert.

	Beschreibung	*Beispiel*
ID	<Cloud Kürzel>-<FortlaufendeNummer>	*AWS-21*
Titel	Sprechender Titel	*Der Datenverkehr von und zu Objektspeichern ist stets zu verschlüsseln*
Kritikalität	Einstufung angelehnt an NIST CVSS 3.0	*HIGH*
Beschreibung	Beschreibung des Sicherheitsrisikos, einem Kontext, der Bedrohungen oder Angreifer, möglicher Auswirkungen. Umfang idealerweise 2-3 Sätze.	*Daten die unverschlüsselt zwischen dem Client und den Cloud-Speicher übertragen werden, können von unbefugten Dritten ausgespäht oder auch unbemerkt verändert werden. Dies kann zu wirtschaftlichen Nachteilen des Unternehmens führen, und/oder die Vertraulichkeit personenbezogener Daten verletzten.*
Sicherheitsempfehlung	Beschreibung welche Cloud-Konfiguration empfehlenswert ist bzw. welche zu vermeiden oder untersagt ist. Die betroffenen Cloud-Ressourcen sind konkrte zu benennen und auf fachliche Konfigurationen hinzuweisen.	*Die Daten sind entwedet vor dem Versand auf seiten des Clients zu verschlüsseln, oder es eine Transportverschlüsselung (TLS) ist zwingend erforderlich. Es wird empfohlen in jedem Fall die Transportverschlüsselung einzusetzen.*
Prüflogik	`Policy as Code Logik mit Abfrage (Selektion/Query) auf die Konfiguration der Cloud-Ressource(n), Entscheidungslogik (wenn/dann) und einem eindeutigen Ergebnis TRUE oder FALSE.`	*abhängig von der gewählten Implementierungsbasis*
Behebung	Entweder eine schrittweise Beschreibung zur Behebung, oder auch `Logikcode`.	*abhängig von der gewählten Implementierungsbasis*
Compliance	Verweise auf relevante Kontrollen bei Sicherheitsstandards.	*NIST 800-53: SC-8 Transmission Confidentiality and Integrity*
Referenzen	Verweise / Hyperlinks auf weiterführende Dokumentation oder Quellen der Sicherheitsempfehlung.	*https://docs.aws.amazon.com/de_de/AmazonS3/latest/userguide/security-best-practices.html*

Bild 6: Fachliche Struktur einer Sicherheitsrichtlinie

3.3.3 Automatisierter Ablauf der Auditierung

Der automatisierte Auditierungsablauf stellt den wesentlichen Schritt der Identifikation der möglichen Schwachstellen dar. Eine CSPM-Lösung nutzt die erhobenen und gegebenenfalls durch den Analyseschritt aufbereiteten Cloud-Konfigurationsdaten und führt diese den Sicherheitsrichtlinien zu. Die Lösung führt die passenden maschinenlesbaren Sicherheitsregeln gegen diese Ist-Zustandsdaten aus und erkennt dadurch Abweichungen vom erwarteten Soll. Es sollte bestimmbar sein, welche Regelsätze für welche Cloud-Konfigurationsdaten zur Anwendung kommen. Es kann beispielsweise durchaus einen Unterschied machen, ob der untersuchte Cloud-Bereich hochsensible Informationen oder Daten aus einer bestimmten Wirtschaftszone umfasst und welche Sicherheitsregeln als Verstöße anzusehen sind. Die gewonnen Erkenntnisse sollten gespeichert werden, damit auf Basis derer ein Ausgabe oder Reaktion zur Mitigation der Schwachstelle erfolgen kann.

Die automatisierte Auditierung sollte durch verschiedene Wege eingeleitet werden können. Zu nennen sind beispielsweise:

- eine zeitlich wiederkehrende Prüfung die im Tages, Wochen, oder Monatstakt erfolgt und dadurch eine kontinuierliche Zustandssicht der aktuellen Sicherheitslage ermöglicht
- auf Basis eines eintretenden Ereignisses, wie der bevorstehenden Installation einer neuen Anwendung und deren Cloud-Infrastruktur-Konfiguration

- einer manuellen Aufforderung zur Sicherheitsprüfung
- dem Eintreten eines Sicherheitsereignisses, das eine automatisierte Prüfung einer Cloud-Infrastruktur zur Folge hat

3.3.4 Ausgabe der Prüfergebnisse und gegebenenfalls der Mitigation

Die Prüfergebnisse eines automatisierten Auditierungsablaufs sind durch eine aussagekräftige Ausgabe zu unterstützen. Die Aussage sollte dabei gefundene Sicherheitsmängel nennen und diesen den Cloud-Ressourcen, die aktuell diese Mängel aufweisen, zuordnen. Die Zuordnung sollte so exakt wie möglich sein um etwaige Verwechslungen zu vermeiden. Sollten die Sicherheitsmängel sich klassifizieren lassen, ist die Kritikalität klar erkennbar darzustellen und sollte sich idealerweise an einem internationalen Standard orientieren.

Falls Maßnahmen zur Behebung des Sicherheitsmangels bekannt sind, sind diese mindestens auszugeben beziehungsweise darauf zu verweisen. Diese Maßnahmen sollten idealerweise eine schrittweise Verfahrensanweisung aufzeigen, um dem Empfänger der Ausgabe eine strukturierte Hilfestellung leisten zu können. Sollten Maßnahmen zur Behebung automatisierbar zur Verfügung stehen und es auch das Ziel der nutzenden Organisation sein diese im jeweiligen Kontext direkt und ohne weitere menschliche Prüfung anzuwenden, sollte diese prinzipielle Möglichkeit bestehen.

Es wäre auch denkbar, dass die Ausgabe der Prüfergebnisse bei identifizierten Sicherheitsmängeln einen Sicherheitsvorfall in einem zentralen Sicherheitssystem auslöst und die Prüfergebnisse darin einfließen.

4 Analyse der Cloud Management-Schnittstellen

4.1 Vorgehensweise

Ein Application Programming Interface (API) - im deutschsprachigen Raum auch als Programmierschnittstelle benannt - ist ein Programmteil eines Softwaresystems der zur geregelten und definierten Kommunikation mit anderen externen Programmen angeboten wird. Die Anbindung erfolgt hier typischerweise auf Quellcode-Ebene, wobei die Programmierschnittstelle durch eine aussagekräftige Dokumentation beschrieben ist. Die Bereitstellung der Integrationsmöglichkeit über das API kann durch Software Development Kits (SDKs), die für die jeweilige Programmiersprache ausgeprägt sind und die notwendigen Bibliotheken bereitstellen, unterstützt werden.

In diesem Kapitel wird das API zur Verwaltung der Cloud-Infrastruktur eines Cloud-Providers strukturiert untersucht, um die technischen Möglichkeiten und Grenzen einer automatisierten Sicherheitsauditierung ermitteln zu können. Als Cloud-Provider wird im Rahmen dieser Arbeit Amazon Web Services (AWS) verwendet. Die Prinzipien lassen sich vermutlich aber auf andere Cloud-Umgebungen wie Microsoft Azure (Azure) oder die Google Cloud Platform (GCP) übertragen.

Methodisch wurden dafür passende *Sicherheitsleitfragen* aufgestellt, um die Cloud-Ressourcen daraufhin zu untersuchen, ob über das API Informationen zu Sicherheitskonfigurationen ermittelt werden können und wie die Rückgabewerte der API-Operationen semantisch zu interpretieren sind. Dies schafft die wesentliche Grundlage, um darauf aufbauend automatisierte Sicherheitsregel entwerfen zu können. Es ist dazu notwendig, die exakten Operationen, deren Attribute oder Schlüssel in der JSON-Notation und Wertebereiche zu ermitteln. Nur auf dieser exakten und detaillierten Ebene können programmatische Regelsätze später greifen.

Die Analyse der Ressourcen wird in einer gleichartigen Form ausgeführt. Auf Basis der *Sicherheitsleitfragen* wird das API strukturiert analysiert, sowie kontextuell

notwendige Informationen der Cloud-Umgebung erhoben und dargestellt. Die Ergebnisse der API-Analyse werden jeweils in einer Tabelle übersichtlich dargestellt, sowie die *Erkenntnisse* in einem Abschnitt zusammengefasst.

Der Untersuchungsraum wurde dabei zunächst auf die für Cloud Security Posture Management (CSPM) relevanten Dienste aus dem Leistungsportfolio der Infrastructure as a Service (IaaS)-Dienste eingeschränkt. Die Untersuchung deckt dabei die elementaren Dienst-Bereiche ab und wird in Dienstgruppen gegliedert:

- Geographische Lage der Ressourcen
- Rechendienste
- Ressourcen zur Datenspeicherung (Block- und Objektspeicher)
- Netzwerkressourcen und Netzwerksicherheitsdienste
- Schlüsselmanagement kryptografischer Schlüssel
- Verwaltung von Identitäten und der Zugriffskontrolle

Der definierte Untersuchungsraum ist in Tabelle 1 aufbereitet. Die zu untersuchenden AWS-Dienste oder Ressourcen sind benannt.

Der Analyse-Ablauf orientiert sich dabei an den Kernfunktionen, die zu einer Verwaltung von Ressourcen zählen, wobei lesende Operationen im Fokus stehen werden. Modifizierende Operationen können im Rahmen der prototypischen Umsetzung exemplarisch zum Einsatz kommen.

IaaS-Funktionsbereich	**AWS Dienstnamen (Kürzel)**
Basisdienste und Strukturen	AWS Account und Organization Region, Verfügbarkeitszone (Availability Zone) und lokale Zone
Rechendienste (Computing)	EC2 Instance, Network-Interface, Volume Amazon Machine Image Instance-Metadatenservice (IMDS)
Datenspeicher und -sicherung	Elastic Block Store Elastic File System Simple Storage Service

IaaS-Funktionsbereich	AWS Dienstnamen (Kürzel)
Netzwerk und -Sicherheit	EC2 Virtual Private Cloud (VPC), Subnet, Route / RouteTable
	Route 53 (DNS)
	Shield (DDoS)
	Internet Gateway
	Virtual Private Network (VPN)
	NetworkAcl
	SecurityGroup
	Network Firewall
	Web Application Firewall (WAFv2)
Schlüsselverwaltung	Key Management Service (KMS)
	EC2 KeyPair
Identitäts- und Zugriffsverwaltung	Identity und Access Management (IAM)
	Access Control List (ACL)

Tabelle 1: Übersicht potentiell CSPM relevante IaaS- und FaaS-Cloud-Dienste von AWS

Die Tabelle 1 stellt den für diese Thesis gewählten Ausschnitt der AWS Dienste dar. Im Anhang A.3 wird in Tabelle 6 eine nach aktuellen Erkenntnissen vollständigere Liste ergänzend bereitgestellt.

Im Glossar befindet sich zur besseren Verständlichkeit eine Kurzbeschreibung der wesentlichen AWS-Dienste, die in der Tabelle 1 aufgeführt sind.

4.2 Analyse der Cloud-Programmierschnittstelle (API)

Der Cloud Service Provider Amazon Web Services stellt für die Verwaltung seiner Cloud-Dienste eine Programmierschnittstelle bereit. Die von der Programmiersprache unabhängige API nennt sich *Query API*. Die Query API nutzt als Schnittstellentechnologie Web-Services, das HTTPS Kommunikationsprotokoll und XML Datenformate. Die umfangreiche Dokumentation der Programmierschnittstelle ist

auf Basis der AWS-Produkte oder -Services strukturiert. Die API-Dokumentation der jeweiligen AWS-Produkte beinhaltet wiederum *Services* als weitere Ebene der Gliederung. Beispielsweise gliedert sich die 2650 Seiten starke API-Dokumentation [24] zu dem Produkt Elastic Cloud Computing (EC2) in 67 Services wie das Amazon Machine Image (AMI) oder Amazon Instances (virtuelle Maschinen). Die API Interaktion erfolgt anhand sogenannter *Actions* die spezifisch zu jedem Dienst bereitgestellt werden. Die Namensgebung der Actions folgt einer gewissen Normierung. Die typischen Grundoperationen Erzeugen-Auslesen-Aktualisieren-Löschen (CRUD - create, read, update, delete) werden in der Regel angeboten und nach dem folgenden Schema benannt: Create<Service>, Describe<Service>, Modify<Service>, Delete<Service>. Weitere *Actions* sind für den jeweiligen Service spezifisch ausgeprägt, so beispielsweise zum Starten und Stoppen virtueller Maschinen *StartInstances* und *StopInstances*.

Das API wird als Abstraktionsschicht der generischen Query API zusätzlich in Form von SDKs für verschiedene Programmiersprachen angeboten. Neben der Verwaltung der Cloud-Dienste über eine grafische Benutzeroberfläche (bei Amazon wird diese als *AWS-Managementkonsole* bezeichnet) läßt sich somit auf programmatische Weise die Cloud-Infrastruktur und deren Ressourcen erzeugen, auslesen, ändern und auch wieder löschen. Die nun folgende Analyse basiert auf dem Query API.

4.2.1 Globale Infrastruktur - Regionen und Verfügbarkeitszonen

Amazon verfügt über eine globale Infrastruktur von Rechenzentren und Netzwerken, die es ermöglicht virtualisierte Maschinen und Dienste zu betreiben. Die Standorte, an denen AWS Rechenzentren unterhält, sind in sogenannte *Regionen* und *Verfügbarkeitszonen (Availability Zones)* gegliedert. Jede Region ist ein separater geografischer Bereich und ist vollständig von anderen Regionen getrennt, um eine größtmögliche Fehlertoleranz und Stabilität zu ermöglichen. Verfügbarkeitszonen sind eigenständige Standorte, die so konzipiert wurden, dass sie von Ausfällen in anderen Verfügbarkeitszonen in derselben Region isoliert sind. Netzwerkverbindungen zwischen den Verfügbarkeitszonen einer Region weisen eine geringe Latenz auf. Regionen bestehen aus einer oder mehreren Availability Zones und sind geografisch verteilt. Ressourcen wie beispielsweise virtuelle Instanzen oder Datenspeicher sind einer Region zugeordnet und werden nicht automatisch über unterschiedliche Regionen repliziert.

Aus Sicht der Informationssicherheit spielt die geographische Verteilung der Rechendienste (Computing) und der verarbeiteten Daten eine wichtige Rolle. Datenschutzrechtliche Normen wie die EU-Datenschutz-Grundverordnung (DSGVO), dem seit 2018 bestehenden US CLOUD Act (Clarifying Lawful Overseas Use of Data) oder auch das relativ neu in Kraft getretene Gesetz zum Datenschutz in China, dem Data Security Law (DSL), sind für international tätige Unternehmen und Organisationen verpflichtend zu beachten [25, 26, 27, 28, 29]. Datenschutzrechtliche Bestimmungen schränken oftmals den geographischen Raum, in dem Daten verarbeitet werden dürfen, ein. Des Weiteren kann eine geographische Verteilung Cloud-gestützter Infrastruktur vorteilhaft sein hinsichtlich des IT-Schutzziels der Verfügbarkeit und somit auch der Resilienz von Anwendungen und der verarbeiten Daten.

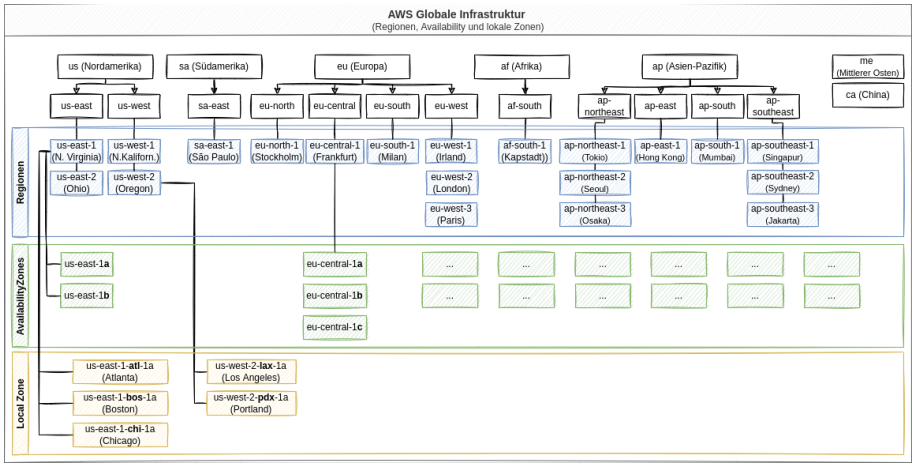

Bild 7: Übersicht AWS Globale Infrastruktur

In Bild 7 ist ein Ausschnitt aus dem derzeitigen Stand der globalen AWS-Infrastruktur schematisch abgebildet. Die technische Benennung der Regionen und Verfügbarkeitszonen folgt einem strikten Muster. Auf oberster Ebene gliedert Amazon die Welt in logische Einheiten die sich weitgehend an den Kontinenten orientieren. In der zweiten Ebene erfolgt eine Gliederung in der Regel nach Himmelsrichtungen. Innerhalb dieser geographischen Einheiten können nun ein oder mehrere Regionen angesiedelt sein, die durch eine fortlaufende Nummerierung eindeutig gekennzeichnet sind. Innerhalb der Regionen können nun ein oder mehrere Verfügbarkeitszonen (Availability Zones) existieren, die technisch durch den Zusatz von Kleinbuchstaben namentlich unterschieden werden. In Nordamerika stellt AWS neben den Verfügbarkeitszonen zusätzlich noch sogenannte *Local Zones* in ausgewählten Regionen bereit. Local Zones sind speziell für latenzempfindliche Anwendungen gedacht und stehen näher an

großen Bevölkerungs- und Industrie-Zentren. Die Benennung der Local Zones erfolgt mittels Erweiterung der Regionenbezeichnung um ein Kürzel aus drei Buchstaben, sowie darauf folgend einer fortlaufenden Ziffer und eines Buchstaben.

Beispielsweise wird das AWS-Rechenzentrum in Frankfurt am Main, Deutschland, mit dem Regionen-Code `eu-central-1` identifiziert. In dieser Region werden derzeit drei Verfügbarkeitszonen angeboten: `eu-central-1`*a*, `eu-central-1`*b* und `eu-central-1`*c*. Eine Local Zone in Atlanta, der Hauptstadt des US-Bundesstaates Georgia, befindet sich in der Region `us-east-1` und wird identifiziert über `us-east-1-`*atl*`-1a`.

Zusammenfassung der Namenskonvention

In Tabelle 2 ist die derzeitig angewandte Konvention bei der Benennung von Regionen, Verfügbarkeitszonen bzw. von Local Zones zusammenfassend dargestellt.

Zone	**Namenskonvention**
Region	[Kürzel Kontinent]-[Himmelsrichtung]-[fortl. Nummer]
Verfügbarkeitszone	[Region][fortl. Buchstabe]
Local Zone	[Region]-[Kürzel 3-Buchstaben]-[fld. Nummer][fortl. Buchstabe]

Tabelle 2: Namenskonvention AWS Regionen und Zonen

Analyse der API-Methoden und Rückgabewerte

Auf Basis obiger Erkenntnis ist zu untersuchen, welche Aufruf-Methoden der API, sowie welche Objekte und Schlüssel oder Attribute in der Rückgabestruktur für eine automatisierte Sicherheits-Auditierung oder Konformitätsprüfung relevant sein können.

Angewendete sicherheitsrelevante Leitfrage(n) für die API-Analyse:

1. Wie kann die geographische Lage der durch den Kunden genutzten Cloud-Ressourcen ermittelt werden?
2. Kann die geographische Lage direkt aus den Rückgabewerten ermittelt werden, oder sind weitere Zusatzinformationen notwendig?

3. Welcher Detaillierungsgrad der geographische Lage kann erreicht werden?

Aus den derzeit circa 600 Actions der Query API für das Produkt EC2 sind in diesem Kontext zwei für die automatisierte Informationserhebung der Cloud-Konfiguration wesentlich. Die Rückgabe der zwei Actions *DescribeAvailabilityZones* und *DescribeRegions* liefert jeweils eine Liste von Regionen und Verfügbarkeitszonen, die generell für den AWS Account nutzbar sind. Die Werte der Schlüssel liefern die einzelnen Identifizierer als Zeichenkette zurück. Das Analyseergebnis ist in Tabelle 3 dargestellt.

API: `EC2.DescribeAvailabilityZones und EC2.DescribeRegions`

Pfad/Objekt	**Schlüssel**	**Werte und CSPM-Relevanz**
Region[]	regionName	Zeichenkette mit dem Bezeichner (Code) der Region, Beispiel `eu-central-1`. Syntax folgt der Aufstellung aus Tabelle 2. Frage(n): 1-3
AvailabilityZone[]	zoneName	Zeichenkette mit dem Bezeichner der Verfügbarkeitszone (Availability Zone), Beispiel `eu-central-1a`. Syntax folgt der Aufstellung aus Tabelle 2. Frage(n): 1-3
	zoneId	Zeichenkette mit der technischen Kennzeichnung der Verfügbarkeitszone. Beispiel: zoneId `euc1-az2` entspricht der Verfügbarkeitszone mit zoneName `eu-central-1a`. Wie erkennbar, lässt sich die technische Kennzeichnung nicht einfach aus dem Namen der Verfügbarkeitszone ableiten (az*2* versus central-1*a*). Frage(n): 1-3
	zoneType	Es werden drei Typen unterschieden: `availability-zone`, `local-zone` und `wavelength-zone`. `Wavelength` ist speziell für 5G-Netzwerk gestützte Anwendungen, wiederum um niedrige Latenz zu erreichen. Frage(n): 3

Tabelle 3: API Objekte - Geographische Information bzw. Rechenzentren

Aufgrund der von AWS gewählten Namenskonvention, wie in Tabelle 2 dargestellt, kann die geographische Lage direkt auf Ebene von Kontinenten ermittelt werden. Aus dem Wert des Schlüssels *regionName* und *zoneName* kann direkt festgestellt werden, ob es sich um eine Region in Europa (eu) oder in den Vereinigten Staaten von Amerika (us) handelt. Wird eine genauere geographische Lage benötigt,

zum Beispiel in welchem Land oder in welchem amerikanischen Bundesstaat Ressourcen verwendet werden, ist eine zusätzliche Zuordnungstabelle zu erstellen. Die Zuordnungstabelle ordnet dabei den Identifizierer der Region den weiteren geographischen Daten zu. Eine entsprechende Zuordnungstabelle wurde erarbeitet und ist im Anhang A.2 verfügbar.

Erkenntnisse Mit den bisher aufgezeigten API-Actions können geographische Informationen abgeleitet werden, jedoch kann noch keine Zuordnung zu den genutzten und konfigurierten Cloud-Ressourcen des Cloud-Kunden erfolgen. Die geografische Lage kann auf Ebene von Kontinenten sowie auf Basis von festgelegten Regionen erfolgen. Über die Regionen-Codes kann, wenn benannt, das Land oder sogar eine Stadt definiert werden.

Als nächsten logischen Schritt bietet sich daher nun an, das API der Ressourcen der Rechendienste (Computing) sowie der Speicherung von Daten zu untersuchen.

4.2.2 Rechendienste und Ressourcen

Der funktionale Schwerpunkt der Elastic Cloud Computing (EC2) ist die Bereitstellung flexibel skalierbarer Rechenkapazität. Diese wird in Form virtualisierter Datenverarbeitungsumgebungen, sogenannter *Instanzen*, bereitgestellt. Dies sind virtuelle Maschinen (VM), die mit unterschiedlichen Betriebssystemen betrieben werden können. Die Einrichtung der Instanzen basiert auf Abbildern für Instanzen. Ein solches Abbild wird bei AWS als *Amazon Machine Image (AMI)* bezeichnet und beinhaltet ein vorkonfiguriertes Betriebssystem sowie übliche Basis-Softwarepakete für den Betrieb in der Cloud (z.B. cloud-init). Ein AMI wird typischerweise auf Basis einer zeitpunktbezogenen Sicherung in Form eines virtuellen Abbildes (Snapshot) einer Instanz und der zugeordneten Speicherlaufwerke aufgebaut. AWS stellt verschiedene AMI auf einem virtuellen Marktplatz (auch AMI Katalog genannt) zur Auswahl bereit. Neben Abbildern, die durch Amazon selbst erstellt und verwaltet werden, können andere Anbieter ebenfalls AMIs erstellen und in dem AMI Katalog zur Verfügung stellen. Der Cloud-Nutzer kann selbst ebenfalls Abbilder erstellen und verwalten. Jedes Abbild ist somit einem Ersteller oder Eigner zugeordnet. AWS unterscheidet hier in AWS-eigene Abbilder, Abbilder von verifizierten Drittanbietern und denen aus der Rubrik 'Community'. In letzterer Kategorie kann jeder ein AMI bereitstellen, ohne jegliche weitere Informationen über die Quelle anzugeben. Verifizierte Drittanbieter scheinen meist namhafte Unternehmen zu sein, wie beispiels-

weise Redhat, Microsoft, oder auch Palo Alto Networks. Die Hürde diesen Status als verifizierter Anbieter zu erlangen, scheint aber aus Sicht der Informationssicherheit nicht besonders hoch zu sein. Amazon schreibt dazu [30]:

> *Um ein verifizierter Anbieter zu werden, müssen Sie sich als Verkäufer auf der AWS Marketplace registrieren. Nach der Registrierung können Sie Ihr AMI auf der AWS Marketplace auflisten.*

Es darf somit *nicht* davon ausgegangen werden, dass die durch Drittanbieter angebotenen AMI-Produkte einer Qualitäts- oder sogar einer Sicherheitsprüfung durch den Cloudanbieter AWS unterzogen werden, wenn der Drittanbieter das Prädikat *Verifizierter Anbieter* trägt.

In diesem Kapitel werden die Dienste aus dem IaaS-Funktionsbereich *Rechendienste (Computing)*, wie in Tabelle 1 aufgeführt, untersucht. Die dazu ausgewählten Fragen zur API-Analyse der EC2-Instanzen sollen zunächst für einen Überblick der Rechendienste in der Cloud-Infrastruktur des Cloud-Kunden bzw. dessen Account sorgen. Verschiedene Untersuchungen und Berichte zur Informationssicherheit und möglicher Risiken in der Cloud weisen als eine grundlegende Problemstellung eine mangelhafte Transparenz sowie fehlende und nicht aktuelle Inventarisierung der Cloud-Ressourcen aus. Diese werden oftmals auch als Cloud-*Assets* bezeichnet. Unvollständiges Wissen zu eingesetzten Betriebssystemen und deren Versionen können sowohl zu nicht erkannten Schwachstellen als auch zu mangelhafter Konformität zu den Sicherheitsrichtlinien oder gesetzlichen Normen führen. Die Erstellung virtueller Instanzen auf Basis nicht geprüfter Abbilder (Images) oder die Nutzung von Images aus nicht vertrauenswürdigen Quellen, können ein Risiko für die Informationssicherheit darstellen. Netzwerk-Konfigurationsfehler bei der Bereitstellung (Deployment) virtueller Maschinen, die zu einer unerwünscht ungeschützten Exposition der IT-Systeme führen, bleiben nicht selten längere Zeit unerkannt und können Angreifern einen initialen Angriffsvektor bieten. So ist auch die Kenntnis in welchen Netzwerksegmenten virtueller Maschinen betrieben werden sowie die Kenntnis über das Vorhandensein und die Konfiguration von IT-Sicherheitssysteme (z.B. Firewalls, oder Intrusion Detection Systemen) für eine automatisierte Analyse und Beurteilung relevant.

Angewendete sicherheitsrelevante Leitfrage(n) für die API-Analyse:

1. Welche Instanzen sind konfiguriert, wie ist deren Zustand (aktiv/passiv) und wie werden diese eindeutig identifiziert?

2. Lassen sich Kontext-Informationen zur Instanz ermitteln? Beispiel: Wer (Benutzer oder Rolle) hat die Instanz angelegt und wann erfolgte dies? Wann wurde die Instanz zuletzt gestartet / gestoppt usw.?
3. Wie kann zur EC2-Instanz die geographische Lage ermittelt werden (Kontinent, Land oder Staat)?
4. Auf Basis welches Abbildes wurde die Virtual Machine (VM) instanziiert? Von wem stammt dieses Abbild?
5. Welches Betriebssystem und welche Version nutzt die VM?
6. Mit welchem Netzwerk ist die Instanz verbunden, welche Netzwerkadresse wird genutzt? Handelt es sich um ein öffentliches, oder ein privates Netzwerk?
7. Welche Datenspeicher sind per Cloud-Konfiguration mit der Instanz verbunden?
8. Ist ein Fernzugriff auf die Instanz aktiviert? Werden dazu kryptographisch gesicherte Verfahren eingesetzt? Wenn ja, welche Schlüssel werden verwendet und welche Eigenschaften weisen diese auf?
9. Wurden Instanz-Metadaten konfiguriert und wenn ja, was ist deren Inhalt?
10. Ist erkennbar, ob der Zugriff auf die VM durch technische Schutzmaßnahmen eingeschränkt wurde (z.B. Netzwerk- oder Host-Firewall)?
11. Kann erkannt werden, ob die Instanz in eine Datensicherung eingebunden ist?

Die Untersuchung der API-Dokumentation zeigt als zentralen Einstieg die Actions `DescribeInstances` und `DescribeInstanceStatus`.

Aus der sehr umfangreichen und geschachtelten Rückgabestruktur der API-Aufrufe wurden die wesentlichen Objekte und Schlüssel herausgearbeitet und in Tabelle 8 (siehe Anhang A.3.3) dokumentiert.

Erkenntnisse Das bisherige Analyse-Ergebnis zeigt, dass einige der Fragestellungen direkt aus den Werten der Schlüssel beantwortet werden können. Beispielsweise kann der Status der Instanz (`instanceState.code`) direkt ermittelt werden. Auch die geographische Lage der Instanzen kann nun ausgewertet werden (`placement.availabilityZone`). Zu anderen Fragestellungen sind Analysen weiterer API-Operationen und Schlüssel/Werte notwendig. Hierzu konnten bereits die weiterführenden Identifizierer ermittelt werden. Beispielsweise kann die verwendete Abbild über den Schlüssel `imageId`

ermittelt werden, es fehlen aber bisher die Informationen zu Quelle oder dem Herausgeber oder Betriebssystem des Abbildes. Auch die von der Instanz genutzten virtuellen Laufwerke können identifiziert werden, indem die Liste `blockDeviceMapping` ausgewertet wird und daraus die `volumeId` als Fremdschlüssel (FK) genutzt wird.

Abbilder für virtuelle Maschinen - Amazon Machine Images

Die bisher genutzte API-Action `DescribeInstances` ermöglicht es zwar zu ermitteln welches Amazon Machine Image (AMI) für die Erzeugung der EC2 Instanz genutzt wurde, jedoch stellt die Action keine weiteren Details zu dem AMI selbst bereit. Für detaillierende Informationen zu den Abbildern ist eine weitere Action namens `DescribeImages` zu nutzen.

Die Analyse der API-Dokumentation zeigt, wie in Tabelle 9 (siehe Anhang A.3.4) aufbereitet, Schlüssel und Werte zu dem Eigner bzw. Ersteller (owner) des Images, sowie potentiell Daten zu dem Betriebssystem. Die Beschreibungen der API-Dokumentation bleibt jedoch zu oberflächlich, welche exakten Informationen und auf welchem Detaillierungsgrad in den jeweiligen Schlüsseln abrufbar sind. Um die noch nicht ausreichenden Informationen der API-Dokumentation zu ergänzen sind ausgewählte Abbilder aus den Amazon Marktplatz über API-Aufrufe zu analysieren.

Das kleine Python-Programm in Listing 4.1 ruft eine Liste von Amazon Machine Images, deren Schlüssel und Werte ab. Die API-Anfrage kann beispielsweise auf einen Anbieter, dessen Amazon Machine Images zu untersuchen sind, mittels eines sogenannten Request-Filters eingegrenzt werden.

```
import boto3, json, pprint
# Filter: 538276064493 ist der Account der Alpine Linux Organisation
filterAlpineLinux = [{'Name':'owner-id', 'Values':['538276064493']}]

# EC2 Client erzeugen und Methode mit Filter aufrufen
client = boto3.client('ec2')
response = client.describe_images(Filters=filterAlpineLinux)
```

Listing 4.1: Programmcode Boto3 AMI Analyse

Die Ausgabe des Aufrufs der Methode ist in Listing 4.2 in gekürzter Form dargestellt. Die gekürzter Form zeigt ein AMI des Herstellers *Alpine Linux Organisation* mit der Eigner ID `538276064493`, sowie eine Auswahl relevanter Schlüssel für eine automatisierte Sicherheitsauditierung.

```
{
```

```
 "ImageId": "ami-0184da2b14a627aa9",
 "OwnerId": "538276064493",
 "PlatformDetails": "Linux/UNIX",
 "State": "available",
 "BlockDeviceMappings": [...],
 "Description": "Alpine Linux 3.15.0 - https://alpinelinux.org/cloud",
 "Name": "alpine-3.15.0-aarch64-uefi-tiny-r3",
 ...
}
```

Listing 4.2: Analyse AMI - Rückgabewerte zu einem AMI der Alpine Linux Organisation

Die Fragestellung Nr. 5 bezügliche des Betriebssystems des Abbilds könnte aus den beschreibenden Schlüsseln `PlatformDetails`, `Name` und `Description` gewonnen werden. Die `OwnerId` identifiziert den Hersteller des Abbilds im Amazon Marktplatz (Frage 4). Der Schlüssel `ImageOwnerAlias` ist in diesem Fall nicht gefüllt.

```
# Auszug eines AMI von Amazon mit Windows Server 2022
{
 "ImageLocation": "amazon/Windows_Server-2022-English-Full-Base-2022.09.14",
 "OwnerId": "801119661308",
 "Platform": "windows",
 "PlatformDetails": "Windows",
 "Description": "Microsoft Windows Server 2022 Full Locale English AMI provided by
    Amazon",
 "ImageOwnerAlias": "amazon",
 "Name": "Windows_Server-2022-English-Full-Base-2022.09.14",
 ...
}
```

Listing 4.3: Analyse AMI - Rückgabewerte zu AMI von Amazon mit Windows Server

Die Analyse eines zweiten AMI aus dem Marktplatz ist in Listing 4.3 dargestellt. Es wurde ein AMI ausgewählt das von Amazon selbst bereitgestellt wird. Es ist erkennbar, dass nun der Schlüssel `ImageOwnerAlias` mit dem Wert `amazon` belegt ist. Aus den Schlüsseln `Platform`, `PlatformDetails` und `Description` können Informationen über das Betriebssystem ermittelt werden. Jedoch sind die Betriebssystem-Informationen nur auf sehr grober Ebene hinterlegt. Eine Normung, um die Daten verlässlich zu analysieren, ist auch nicht dokumentiert. Es wird zwar die Windows Server Hauptversion ausgegeben, aber keine Informationen über einen Service-Pack Stand oder über bereits installierten Security-Patches.

Zu beachten ist auch, dass eine auf Basis eines Amazon Machine Image (AMI) erzeugte EC2 Instanz sich zwar zum Zeitpunkt der Erstellung hinsichtlich des Softwarestandes und der Konfiguration des Betriebssystems mit dem des Abbild gleichen kann. Eine Aktualisierung des AMI führt jedoch nicht zu einer Aktualisierung der erzeugten Instanz. Wie im *Modell der geteilten Verantwortung* in Kapitel 2.3 bereits

dargestellt, liegt die Verantwortung zur Wartung des Betriebssystems der Instanz bei dem Cloud-Kunden und nicht bei dem Cloud-Anbieter.

Erkenntnisse Die Query API ermöglicht es die Abbilder (englisch Image) einer EC2 Instanz zu ermitteln. Insbesondere zum Zeitpunkt der Erstellung, bzw. idealerweise vor der Erstellung, könnten sicherheitsrelevante Aspekte gewonnen werden. So kann ermittelt werden, ob das Abbild von einem vertrauenswürdigen Anbieter stammt. Die Definition welche Anbieter als vertrauenswürdig zu erachten sind ist jedoch individuell zu klären, da die Klassifizierung des Cloud Service Provider (CSP) AWS alleine nicht ausreichend ist. Inhaltliche Informationen, wie beispielsweise das Betriebssystem, dessen Aktualität hinsichtlich eingespielter Systemaktualisierungen oder Sicherheits-Patches, oder einer durchgeführten und dokumentierten Systemhärtung, können nur rudimentär bis gar nicht ermittelt werden.

Sicherer Fernzugriff auf die Instanz

Der Fernzugriff auf die Kommandozeile einer VM hat typischerweise über eine abgesicherte und verschlüsselte Kommunikationsverbindung zu erfolgen. Unsichere Protokolle wie telnet sind nicht mehr Stand der Technik. Das kryptographische Secure Shell (SSH)-Protokoll ist schon seit vielen Jahren der Standard unixoider Systeme und wird mittlerweile auch in aktuelleren Windows-Versionen unterstützt. SSH nutzt symmetrische Verschlüsselungsverfahren, um die gesamte Kommunikation der Sitzung auf der Transportschicht abzusichern. SSH unterstützt asymmetrische kryptographische Verfahren für die Authentifizierung, sogenannte Public-Key-Authentifizierung, bei der eine Kombination aus einem privaten und öffentlichen Schlüssel benötigt wird.

Aus dem obigen Ergebnis der Query API-Analyse, siehe Tabelle 8 (siehe Anhang A.3.3), geht der JSON-Schlüssel `keyName` als ein Verweis auf ein durch AWS verwaltetes kryptographisches Schlüsselpaar hervor. Der öffentliche Schlüssel wird bei der Initialisierung einer EC2 Instanz auf diese übertragen und gespeichert (Datei: `~/.ssh/authorized_keys`), sodass sich der SSH-Client unter Verwendung des privaten Schlüssels an dem SSH-Server authentifizieren kann. Kryptographische Schlüsselpaare können bei AWS entweder generiert werden, oder es können extern erzeugte Schlüsselpaare importiert werden. Der öffentliche Schlüssel wird bei AWS gespeichert, der private Schlüssel ist an einem sicheren Ort zu verwahren. Kostenpflichtige Dienste von AWS, wie der AWS Key Management Service (KMS) und verwaltete Hardware Security

Module (HSM), können zur sicheren Verwahrung genutzt werden, sind aber nicht verpflichtend.

Um weiterführende Informationen über ein kryptographisches Schlüsselpaar zu erhalten, ist die Verwendung der Query API Action `DescribeKeyPairs`, mit dem `keyName` als Filteroption, notwendig. Der durch den Aufruf der Action zurückgelieferte Datentyp `KeyPairInfo` stellt laut API-Dokumentation folgende Schlüssel und Werte bereit (Auswahl):

- `keyPairId` und `keyName`: Primärschlüssel und frei zu vergebender Name
- `keyType`: kryptographisches Verfahren.
- `publicKey`: der öffentliche Schlüssel.
- `tagSet`: soweit genutzt, frei wählbare Kennzeichnungen.
- `keyFingerprint`: Der Fingerabdruck des privaten oder öffentlichen Schlüssels, in Abhängigkeit des Verfahrens und Art der Erzeugung.

Die Analyse der API-Dokumentation zeigt, dass das verwendete kryptographische Verfahren ermittelt werden kann. AWS unterstützt die Verfahren Rivest–Shamir–Adleman (RSA) und ED25519, das auf elliptischen Kurven basiert. Weiterführende Informationen, wie beispielsweise die verwendete Schlüssellänge, ergibt sich nicht aus der Query API-Dokumentation. Der öffentliche Schlüssel sollte ebenfalls zu beziehen sein. Zur weiteren Analyse der API-Fähigkeiten und der konkreten Rückgabewerte wurden drei Schlüsselpaare erstellt und diese über ein API-Python Programm abgerufen.

1. AWS generiertes RSA Schlüsselpaar mit Namen `kp-masterthesis`
2. extern generiertes RSA Schlüsselpaar mit einer Schlüssellänge von *4096 Bit* und dem Namen `mt-rsa-4096`. Dem Schlüsselpaar wurden beim Import frei gewählte Kennzeichen (Tags) hinzugefügt.
3. extern generiertes ED25519 Schlüsselpaar mit dem Namen `mt-ed25519`. Auch hier wurde ein frei gewähltes Kennzeichen (Tag) hinzugefügt.

Die Ausgabe des Python Programmes zeigt, dass nur das verwendete kryptographische Verfahren ermittelt werden kann, Schlüssellängen jedoch nicht ausgegeben werden (siehe Bild 8, rechte Seite). Aus der AWS EC2 Dienst-Dokumentation geht hervor, dass die derzeit über AWS erzeugten RSA Schlüssel eine Schlüssellänge von 2048 Bit aufweisen und dies auch nicht mittels Konfiguration geändert werden kann.

Dies lässt sich jedoch in der API-Ausgabe nicht differenzieren. Weder ist erkennbar, ob der Schlüssel importiert oder erzeugt wurde, noch sind die jeweiligen Schlüssellängen erkennbar. Auffällig ist weiterhin, dass der öffentliche Schlüssel, obwohl in der aktuellen API-Dokumentation angeben, nicht über den JSON-Schlüssel `publicKey` bereitgestellt wird.

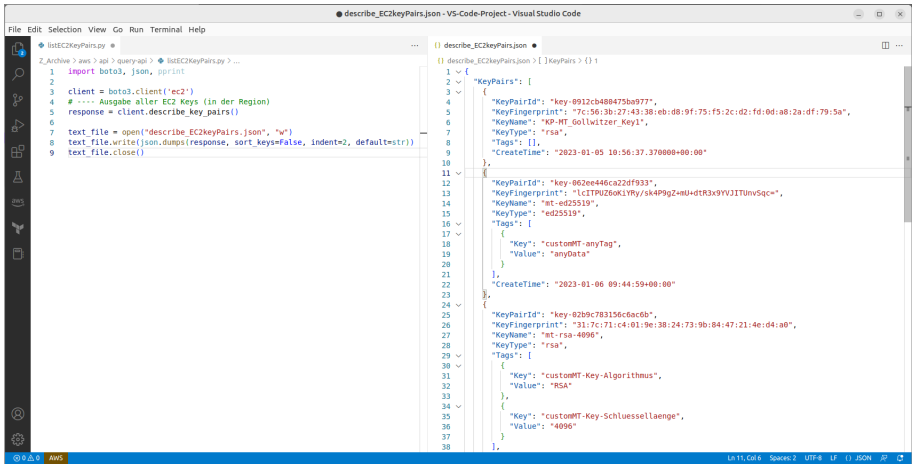

Bild 8: API-Analyse von Schlüsselpaaren für eine gesicherte Kommunikation

Kritischer Blick Das AWS Standardverfahren bei der Erstellung von unixoiden Systemen auf EC2 Instanzen basiert auf der SSH Public-Key Authentifizierung. Diese kann aber durch klassische Konfiguration des SSH-Daemon auf dem Linux-System geändert werden. Setzt man in der Konfigurationsdatei `/etc/ssh/sshd_config` den Parameter `PasswordAuthentication yes` wird eine Anmeldung mittels Passwort erlaubt. Auch kann das Passwort für den privilegierten Linux-User geändert werden. Die Vergabe von einfachen und unsicheren Passworten ist möglich. Eine derart geänderte SSH-Dienst- und Systemkonfiguration ermöglicht einem Angreifer Brute Force Techniken einzusetzen (siehe T1110 in [31]). Beispielsweise nutzt die Malware Kinsing diese Technik um sich über SSH Zugriff auf Systeme und Container zu verschaffen und damit im Netzwerk zu verbreiten [32].

Erkenntnisse Durch eine API-basierte und automatisierte Analyse kann die Möglichkeit zur Verwendung von kryptographischen Verfahren bei dem Fernzugriff grundsätzlich feststellen. Eine Änderung der SSH-Server Konfiguration, wie oben skizziert, kann über das API jedoch nicht geprüft werden! Die verwendeten kryptographischen Verfahren und Schlüssel können der Instanz prinzipiell zugeordnet werden. Sicher-

heitsvorgaben und -empfehlungen bezüglich der Schlüssellängen, wie diese beispielsweise durch das BSI in den *Technischen Richtlinien* veröffentlicht werden, können nicht durch eine automatisierte Auditierung geprüft werden [33, 34]. Aus welchen Netzen der Fernzugriff zugelassen wird ist bisher nicht ersichtlich. Dies ist durch weitere API-Analysen zu prüfen.

Instanz-Metadaten

Metadaten stellen typischerweise strukturierte und maschinenlesbare Informationen und Merkmale zu anderen Daten bereit. Die führenden kommerziellen Cloud-Anbieter, als auch OpenSource-Alternativen, stellen heutzutage einen zentralen Dienst bereit, der Metadaten zu Instanzen der virtuellen Maschinen verwaltet und über eine Schnittstelle zur Verfügung stellt. Dies ermöglicht den virtuellen Maschinen auf, oft dynamisch zugewiesene, Konfigurationsdaten zuzugreifen und diese im Rahmen der Initialisierung und Konfiguration zu nutzen. Beispielsweise kann ein Instanz-Metadaten-Service (IMDS) Informationen wie die IP-Adresse, den Hostname, Informationen zur Region in der die Instanz gestartet wurde, oder auch Zugangsdaten und zugewiesene Berechtigungsrollen bereitstellen.

Instanz-Metadaten werden durch systemnahe Cloud-Standardsoftware, wie beispielsweise *cloud-init*, ausgewertet und können zur Konfiguration des Systems genutzt werden. Instanz-Metadaten können auch von kundenspezifischen Anwendungen, die auf einer VM installiert sind, abgerufen und ausgewertet werden. Dies hilft unter anderem hartkodierte Zugangsdaten im Klartext zu vermeiden, beziehungsweise unterstützt eine effiziente Verwaltung und Verteilung rotierender Zugangs- und Passwortdaten. Durch Funktionen des Instanz-Metadaten-Service (IMDS) ist es auch möglich Berechtigungen auf andere Cloud-Ressourcen über eine rollenbasierte Zugriffskontrolle zu vergeben, ohne dies fest parametrisiert in einer Anwendungskonfiguration oder im Binärcode einer Anwendung auf der Instanz hinterlegen zu müssen. Dies kann die Sicherheit des IT-Systems steigern.

Ein weiterer Bereich der Instanz-Metadaten wird als *User-Data*, oder auch als *Gastattribute* in der Google Cloud Platform (GCP), bezeichnet. Dieser Datenbereich ist persistent und kann von außerhalb der VM zur Laufzeit aktualisiert werden. User-Data wird in der Regel nicht verschlüsselt und kann von jedem Prozess einer VM abgerufen werden. Der Speicherplatz ist mit Größen zwischen 16KB bei AWS und 64KB bei Microsoft Azure (Azure) relativ begrenzt. Ein Anwendungsfall ist

beispielsweise in diesem Datenbereich Shell-Skripte zu hinterlegen, die später zyklisch oder zu bestimmten Ereignissen ausgeführt werden. Die Skripte werden mit privilegierten Berechtigungen ausgeführt. Unter UNIX-Derivaten erfolgt die Ausführung als root-Benutzer. Dieser Datenbereich wird inhaltlich nicht durch den CSP eingeschränkt oder gar sicherheitstechnisch verifiziert.

Der Inhalt des Datenbereichs User-Data kann ein Sicherheitsrisiko darstellen, wenn dort schützenswerte Informationen im Klartext gespeichert werden, oder schädliche Routinen in die virtuelle Maschine eingeschleust würden. Sensitive und schützenswerte Informationen würden unverschlüsselt gespeichert werden, sowie wäre der Zugriff innerhalb der VM auf diese Daten nicht auf wenige Prozesse oder Anwendungen eingeschränkt.

Angriffstechniken um Informationen aus dem Instanz-Metadaten-Service (IMDS) zu erspähen beruhen meist auf Schwachstellen oder Konfigurationsmängeln webbasierten Anwendungen, die auf der VM installiert sind, oder vorgelagerter Komponenten wie Firewalls oder Proxies. Die unter dem Titel *Server Side Request Forgery (SSRF)* bekannte Angriffstechnik, bei der ein Angreifer mittels präparierter URLs die Funktionalität eines Servers zum Auslesen von internen Ressourcen und Daten manipuliert, könnte eingesetzt werden. Das auf die Informationssicherheit von webbasierten Anwendungen spezialisierte Open Web Application Security Project (OWASP), als auch die Artikel von MacCarthaigh und dem Sicherheitsforscher Frichette zur Sicherheit von Instanz-Metadaten bei AWS, beschreiben diese Risiken und Angriffstechniken detaillierter [35, 36, 37].

Der Cloud-Anbieter Amazon Web Services stellt den *Instance Metadata Service* für die Verwaltung von Metadaten bereit. Amazon bietet zwei Versionen seines Dienstes an: IMDSv1 und seit 2019 IMDSv2. Standardmäßig kann bei einer EC2 Instanz entweder IMDSv1 oder IMDSv2 oder beides verwendet werden. IMDSv2 verwendet sitzungsorientierte Anfragen, während IMDSv1 ein reines Anfrage/Antwort-Verfahren unterstützt. Bei der neueren Version erfolgt eine Sitzungsauthorisierung und es wird ein zeitlich begrenztes Sitzungs-Token erstellt.

IMDSv2 gilt als vorteilhaft für eine verbesserte Informationssicherheit und wird empfohlen. Beispielsweise lehnt IMDSv2 `PUT`-Anfragen ab, die einen `X-Forwarded-For` HTTP-Header beinhalten. Dieser Header-Tag wird typischerweise von Reverse-Proxies eingefügt und beinhaltet die ursprüngliche Absenderadresse. Dieser Header zeigt an, dass die Anfrage weitergeleitet wurde und eventuell nicht von einem autorisierten Nutzer stammt. Aus Sicherheitsgründen werden Anfragen dieser Art von

IMDSv2 abgelehnt.

Auf Basis dieses Kenntnisstandes wird die gestellte *sicherheitsrelevante Leitfrage für die API-Analyse* hinsichtlich der Metadaten-Verwaltung weiter verfeinert:

9. Wurden Instanz-Metadaten konfiguriert und wenn ja, was ist deren Inhalt?
   a) Ist der Metadaten Dienst für die Instanz aktiviert?
   b) Welche Instance Metadata Service Version ist erlaubt?
   c) Wurde der EC2 Instanz eine IAM-Rolle zugewiesen, die über den Metadatendienst abgerufen werden könnte?
   d) Wurden Benutzerdaten (User Data) in den Metadaten hinterlegt?

Das Query API verfügt über keine spezifische Action zur Analyse der Instanz-Metadaten. Die Informationen über die Konfiguration der Metadaten einer EC2 Instanz werden über die bereits in Tabelle 8 (Anhang A.3.3) genutzte API-Action `DescribeInstances` untersucht. Der Datentyp `Instance` verfügt über einen Schlüssel namens `metadataOptions` für die Metadaten und deren Konfiguration. Die Metadaten-Optionen werden als eigener komplexer Datentyp `InstanceMetaefdataOptionsResponse` repräsentiert. Für den Zugriff auf den Datenbereich User-Data ist eine zweite Action namens `DescribeInstanceAttribute` zu nutzen.

Erkenntnisse Die in Tabelle 10 (siehe Anhang A.3.5) zusammengefassten Ergebnisse der API-Analyse zeigen, dass wesentlichen Fragestellungen zur Verwendung von Instanz-Metadaten durch die identifizierten Actions und Datentypen automatisiert ermittelt werden könnten. Der lesende Zugriff auf die Instanz-Metadaten selbst scheint jedoch durch die Query API, die außerhalb der EC2 Instanz arbeitet, derzeit nicht möglich zu sein. Ausnahme bilden die Instanz-Metadaten aus dem Bereich User-Data, die lesenden und modifizierenden Zugriff erlauben. Für einen schreibenden Zugriff, zum Beispiel um eine korrigierende automatisierte Sicherheitsmaßnahme durchzuführen, wäre zusätzlich die Action `ModifyInstanceAttribute` einzusetzen. Weiterhin ist es über den Schlüssel `IamInstanceProfile` möglich zu ermitteln, ob der EC2 Instanz eine Berechtigungsrolle zugewiesen wurde. Ob diese Zuweisung fachlich korrekt ist, ist im konkreten Kontext zu definieren.

Zusammenfassung bisherige Analyseergebnisse

Die in diesem Abschnitt durchgeführte API-Analyse der *Rechendienste und Ressourcen* aus dem Infrastructure as a Service (IaaS) Leistungsportfolio konnte bereits Erkenntnisse liefern, welche sicherheitsrelevanten Informationen über die Query API automatisiert ermittelt, ausgewertet und gegen Sicherheitsvorgaben geprüft werden könnten. Erkennbar ist aber auch, dass das API in gewissen Bereichen Grenzen aufweist. So können detaillierte und aktuelle Informationen zu den verwendeten Betriebssystemen der Abbilder nicht ermittelt werden. Der Informationszugriff über das API zeigt hier eher eine Blackbox-Sicht auf die virtuelle Maschine, ohne den inneren Aufbau wiederzugeben. Aus Sicht der Informationssicherheit wären hier andere Vorgehensweisen der Systemauditierung, die direkt auf der virtuellen Maschine operieren, ergänzend einzusetzen. Auch im Bereich kryptografischer Verfahren für den Fernzugriff wurden Defizite erkannt.

Leitfragen mit Bezug zur Netzwerksicherheit verlangen weitere Analyseschritte. Technische Schlüssel zu weiterführenden API-Objekten, wie beispielsweise zu Security-Groups, wurden ermittelt und sind genauer zu untersuchen. Dies gilt auch für konfigurierte und angebundene Datenspeicher.

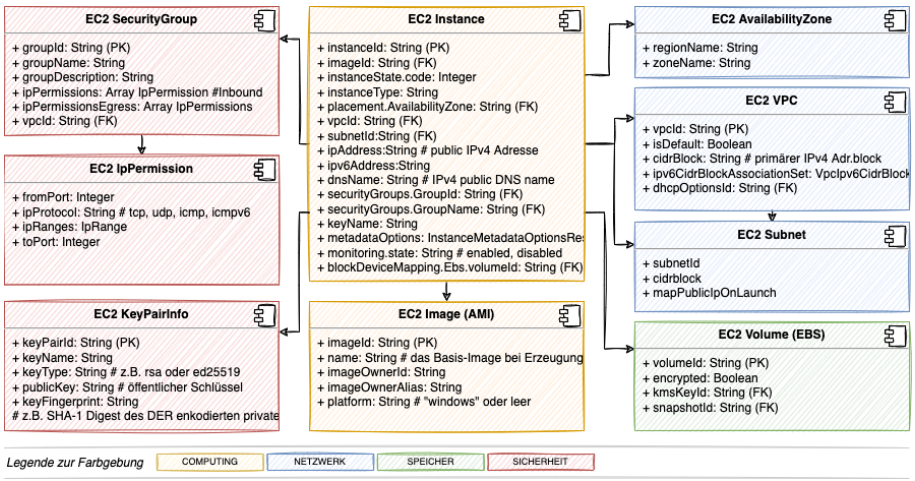

Bild 9: Relationen ausgewählter AWS EC2 API-Objekte

Eine erste zusammenfassende Darstellung der bisherigen API-Analyse der *Rechendienste und Ressourcen* veranschaulicht die Ergebnisse (Bild 9). In die Darstellung sind bereits auch erste Erkenntnisse aus den Folgekapiteln eingeflossen.

4.2.3 Datenspeicher

In diesem Kapitel werden die AWS Speicherdienste überprüft. Es sind zwei Speicherarchitekturen zu unterscheiden. Einerseits gibt es die klassischen Dateisysteme mit Blockspeicherung, die über die AWS-Dienste Elastic Block Store und der Netzwerkdateispeicher Elastic File System, bereitgestellt werden. Die zweite wesentliche Speicherarchitektur ist der Cloud-Objektspeicher bei dem die Daten in Form von Objekten und zugehörigen Metadaten verwaltet werden.

Blockspeicher

Als Blockspeicher dient bei AWS der Dienst Elastic Block Store (EBS). Dieser Dienst stellt virtualisierte Datenträger (englisch Volumes) bereit, die mit EC2 Instanzen genutzt werden können und sich wie unformatierte Blockgeräte verhalten. Diese Speicher können auch unabhängig von dem Lebenszyklus einer EC2 Instanz bestehen. Die Blockspeicher können mit unterschiedlichen Dateisystemen formatiert werden. Von EBS Datenträgern können Abbilder, sogenannte Snapshots, erzeugt werden, die als Datensicherung genutzt werden können, oder als auch als Vorlage für neue EBS-Datenträger dienen.

Diese logischen Datenträger werden in der AWS Infrastruktur erstellt und der EBS-Service stellt sicher, dass die Datenträger vor jeder (Wieder-)Verwendung durch einen Kunden logisch leer sind (d. h. die Rohblöcke werden auf Null gesetzt oder enthalten kryptografische pseudozufällige Daten).

Der EBS Dienst stellte die virtualisierten Datenträger in einer ausgewählten Verfügbarkeitszone (englisch Availability Zone) und somit an einem geographischen Ort bereit. Dies kann aus Sicht des Datenschutzes relevant sein.

Das API für den Elastic Block Store (EBS)-Dienst wird anhand ausgewählter Fragestellungen untersucht.

Angewendete sicherheitsrelevante Leitfrage(n) für die API-Analyse:

1. Wie kann erkannt werden, ob die Daten im Blockspeicher im Ruhezustand verschlüsselt werden?
2. Gibt es Blockspeicher die keiner EC2 Instanz zugeordnet sind (und eventuell vergessen wurden)?
3. In welcher geographischen Region sind die Daten des Blockspeichers?

4. Wie kann erkannt werden, ob eine Datensicherung in Form eines Snapshots verschlüsselt wurde?
5. Kann der Zugriff auf EBS-Datenträger oder auch Snapshots beschränkt werden?

Die Verschlüsselung von EBS-Datenträger kann nur während der Erstellung geschehen. Soll ein bestehender Datenträger nachträglich verschlüsselt werden ist zunächst ein Abbild zu erstellen und auf dessen Basis wird ein neuer verschlüsselter Datenträger erzeugt. Die Verschlüsselung des EBS-Datenträgers wird durch die Aktivierung einer Konfigurationseinstellung erreicht. Wird eine Kopie eines Datenträgers erzeugt, also ein Snapshot, ist dieser stets auch verschlüsselt. Es ist nicht möglich aus einem verschlüsselten Datenträger einen unverschlüsselten Snapshot zu erzeugen. Eine Verschlüsselung eines Datenträger, oder eines Snapshots, kann nachträglich auch nicht entfernt werden. Dies schließt auch ein, dass ein aus einem verschlüsselten Snapshot erzeugter neuer Datenträger stets verschlüsselt ist.

Ein Snapshot kann sensitive Daten enthalten und der Zugriff auf dieses Abbild sollte geschützt werden, bzw. der Zugriff drauf kontrolliert sein, wenn ein Snapshot geteilt werden soll.

Erkenntnisse Die Ergebnisse der Analyse des EBS API sind in Tabelle 11 (siehe Anhang A.3.6) zusammengefasst. Ergänzend zu den Sicherheitskonfigurationen die auf Ebene der einzelnen Datenträger eingestellt werden besteht die Möglichkeit über die API-Action `EnableEbsEncryptionByDefault` die Verschlüsselung als Standard für alle (neuen) virtuellen EBS-Datenträger innerhalb des Kundenkontos (Account) zu aktivieren. Diese Konfigurationseinstellung kann über die Action `GetEbsEncryptionByDefault` ausgelesen werden. Eine Sicherheitseinschränkung, wer berechtigt ist auf Basis des Snapshot ein eigenes Volume zu erzeugen und damit die Daten zu kopieren, konnte in der bisherigen Query-API Analyse nicht identifiziert werden.

Netzwerkdateisystem

Der AWS Dienst Elastic File System stellt ein skalierbares Netzwerkdateisystem für Linux- und Mac-Systeme bereit. Der Elastic File System (EFS) Dienst kann über das Network File System (NFS) Protokoll (NFSv4) genutzt werden, oder über einen eigenen Client-Adapter, den sogenannten *EFS-Mount-Helper*, sowie über das API

von EFS angesprochen werden. Die API-Analyse des EFS Dienstes konzentriert sich auf vier Schwerpunkte.

Angewendete sicherheitsrelevante Leitfrage(n) für die API-Analyse:

1. Wie kann erkannt werden, ob die Daten im Netzwerkspeicher im Ruhezustand verschlüsselt werden?
2. Wie kann der Datenverkehr auf dem Transportweg abgesichert werden und kann der Zugriff auf Netzwerkebene kontrolliert werden?
3. In welcher geographischen Region sind die Daten des Netzwerkspeichers?
4. Kann der Zugriff auf den Netzwerkspeicher durch Autorisierung geschützt werden?
5. Kann der Zugriff auf Ebene des Netzwerks eingeschränkt werden?

Die Verschlüsselung eines EFS-Dateisystems geschieht analog wie bei dem Blockspeicher EBS durch die Konfigurationsdirektive `Encrypted`, die bei der Erstellung des Netzwerkspeichers angegeben werden kann. Eine nachträgliche Verschlüsselung des bestehenden Dateisystems wird, wie bei EBS, nicht unterstützt.

Die Verschlüsselung der Daten während der Übertragung ist abhängig vom gewählten Zugriffsverfahren. Beim Einsatz des Amazon EFS-Mount-Helper wird die Verschlüsselung der Daten über Transport Layer Security (TLS) bereits unterstützt. Der EFS-Mount-Helper nutzt hierzu im Hintergrund das OpenSource Werkzeug *stunnel*, das als transparenter Proxy zwischen Client und Server agiert und einen verschlüsselten Datenkanal für das NFS Protokoll aufbaut. Nutzt man den EFS-Mount-Helper nicht, kann man den gleichen Ansatz verwenden, muss die Konfiguration jedoch selbst aufbauen. Der AWS Dienst stellt auf Port `2049` (Standard NFS) sowohl den unverschlüsselten, als auch den mit TLS abgesicherten Kanal über TCP bereit.

Das Dateisystem kann über den DNS Namen angesprochen werden. Die Notation ist dabei wie folgt: `<FileSystemId>.efs.<RegionCode>.amazonaws.com`. Ein EFS-Netzwerkdateisystem wird stets in einer geografischen Region angelegt.

Ein mit den Standard-Einstellungen erstelltes EFS-Dateisystem kann von jedem EC2-System innerhalb der Cloud-Netzwerkumgebung (Virtual Private Cloud (VPC)) des Mandanten gemountet werden. Es findet keine Einschränkung des aufrufenden Systems statt. Der root-Benutzer (ID 0) des EC2-Betriebssystems bekommt uneingeschränkten, lesenden und schreibenden, Zugriff auf alle Dateien im Dateisystem. Bei

dem NFS-mount Zugriff ist auch keine Autorisierung notwendig und die Transportverschlüsselung wird nicht erzwungen. Um diese potentiellen Sicherheitsrisiken zu

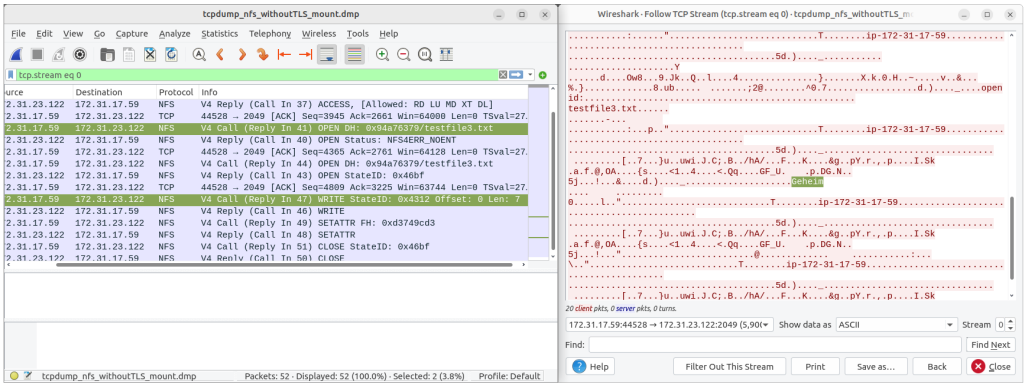

Bild 10: TCP Dump eines unverschlüsselten Zugriffs auf ein EFS-Dateisystem

verifizieren, wurde ein EFS-System mit Standardeinstellungen erzeugt und eine EC2-Instanz auf Basis Alpine-Linux 3.17 erstellt. Das EFS-Dateisystem wurden auf dem EC2-System mit dem root-Nutzer gemountet und der Netzwerkdatenverkehr aufgezeichnet. Es wurde der Mount-Vorgang und die Erstellung einer Textdatei mit dem Inhalt *'Geheim'* mitgeschnitten. Das Bild 10 zeigt den Datenmitschnitt zwischen dem EC2-Client (IP 172.31.17.59) und dem EFS Mount-Target (IP 172.31.17.122). Sowohl der Dateiname (testfile3.txt), als auch der Dateninhalt sind im Klartext erkennbar (siehe Markierungen). Der TCP-Dump im Werkzeug Wireshark zeigt auch, dass keine Autorisierung erfolgt.

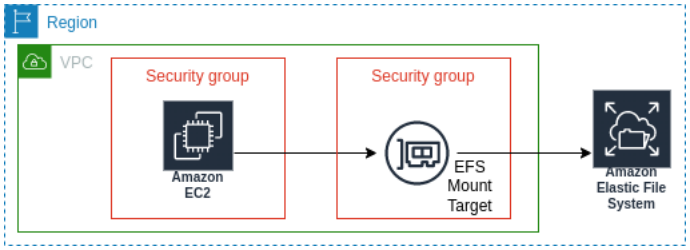

Bild 11: Architekturschema: Zugriff auf EFS-Dateisystem per EFS Mount-Target

Das Architekturschema in Bild 11 zeigt den prinzipiellen Aufbau der Kommunikation. Das EFS-Dateisystem wird über eine spezielle AWS Netzwerkkomponente, das EFS Mount-Target, angesprochen. Das EFS-Dateisystem selbst ist Teil eines zentralen Speichers der sich in der gleichen Region wie das aufrufende Quellsystem, hier die EC2 Instanz, befindet. Das Bild veranschaulicht auch bereits einen Ansatz, wie

das Dateisystem auf Ebene der Netzwerkadressierung geschützt werden könnte. Die *Security Group* ist eine Art Firewall, die den ein- und ausgehenden Netzwerkverkehr zu einer Ressource steuern kann.

Die Existenz, die Zugehörigkeit zu einer Ressource und die Konfiguration, die Firewall-Regeln, einer Security Group kann über das API ermittelt und analysiert werden. Diese Firewall-Regeln zu ermitteln verlangt aber bereits eine komplexere Kette von API Aufrufen und somit die Verkettung von AWS-Objekten und deren Wert-Schlüsselpaaren. Der logische Ablauf der API-Aufrufe ist dabei wie folgt:

1. `DescribeFileSystems` liefert den Identifizierer `FileSystemId` des EFS-Dateisystems.
2. `DescribeMountTargets` liefert zur `FileSystemId` die zugehörigen Mount-Targets und deren Identifizierer `MountTargetId`
3. `DescribeMountTargetSecurityGroups` liefert zur `MountTargetId` eine Liste der zugehörigen Security Groups und deren ID `GroupId`.
4. `DescribeSecurityGroups` liefert nun zur jeweiligen Security Group die Details mit den ein- und ausgehenden Firewall-Regeln.

Auf Basis dieser Abfolge kann durch einen automatisierten Cloud Security Posture Management (CSPM)-Ansatz die Absicherung des EFS-Dateisystem auf Netzwerkebene geprüft werden.

Ausstehend ist jedoch noch die automatisierte Erkennung, ob der Zugriff auf das EFS-Dateisystem durch eine Autorisierung eingeschränkt und geschützt wird. Die Berechtigungskontrolle erfolgt auf Basis von Policies in Verbindung mit dem AWS Identity und Access Management (IAM). Die Dateisystem-Policy steuert dabei vier wesentliche Elemente:

1. Welche Ressourcen werden durch die Sicherheits-Policy gesteuert?
2. Welche Aktionen werden geregelt?
3. Werden die Aktionen erlaubt oder unterbunden, welchen Effekt hat die Regel?
4. Welche Identitäten oder Rollen wirkt sich die Policy aus (auch die Identität 'Alle')?

Wie bereits erwähnt arbeitet die EFS Standard-Policy ohne Authentifizierung und lässt einem anonymen Benutzer alle Rechte. Zur Einschränkung dieser Rechte gibt es drei spezifische Aktionen für den zugreifenden (mountenden) Client:

`elasticfilesystem:ClientMount` erlaubt das mounten des Dateisystems und gibt lesende Berechtigung. `elasticfilesystem:ClientWrite` fügt die Berechtigung hinzu auch Daten schreiben zu dürfen. Um den root-Zugriff auf das Dateisystem zu steuern, kann die Aktion `elasticfilesystem:ClientRootAccess` unterbunden oder vergeben werden.

```
1  {
2    "Statement": [
3      {
4        "Sid": "efs-policy-mt-vorstand-lesen",
5        "Effect": "Allow",
6        "Principal": {
7          "AWS": "arn:aws:iam::126164056430:user/vorstand"
8        },
9        "Action": "elasticfilesystem:ClientMount",
10       "Resource": "arn:aws:elasticfilesystem:eu-central-1:126164056430:file-system/fs-0
     e5f5eaca8c7041a9",
11     }
12   ]
13 }
```

Listing 4.4: EFS Policy die lesenden Zugriff dem Benutzer 'Vorstand' erlaubt

Das Listing 4.4 zeigt eine Policy die dem Benutzer (Principal) *Vorstand* (Zeile 7) die Aktion lesender Zugriff (Zeile 9) erlaubt (Zeile 5) und sich auf ein EFS-Dateisystem bezieht (Zeile 10).

Die Ergebnisse der API-Analyse sind in Tabelle 12 (siehe Anhang A.3.7) zusammengefasst dargestellt. Wichtig ist zu verstehen, dass die Sicherheit des Dateisystems in dem EFS-Dateisystem nicht durch das API analysiert werden kann. Die Dateizugriffsrechte werden, wie bei anderen unixoiden-Dateisystemen geregelt und sind nicht durch das API sichtbar.

Erkenntnisse Die API-Analyse für den Amazon Blockspeicher EBS, sowie das blockorientierte Netzwerkdateisystem EFS, lieferten Zugriff auf essentielle Informationen zur Sicherheitskonfiguration der Dienste. Bewusst muss aber auch werden, dass die administrativen Berechtigungen mit Zugriff auf das Einhängen eines EBS Datenträgers in EC2 Instanzen ein essentielle Sicherheitsberechtigung darstellt. Denn eine server-seitige Verschlüsselung der Daten hilft bei eingehängtem EBS-Datenträger nicht mehr gegen unbefugten Zugriff. Die Berechtigungsverwaltung der Daten auf Ebene des Datei- und Betriebssystems ist durch die API nicht einsehbar oder zu ändern. Hierzu sind klassische Maßnahmen, wie die Härtungen des Betriebssystems und das Berechtigungsmanagement des Dateisystems, zielführend. Es wurde auch festgestellt, dass nun nicht mehr nur die Auswertung der Schlüssel

und Werte eines AWS-Objektes ausreicht, um sicherheitsrelevante Schwachstellen in der Cloud-Konfiguration zu ermitteln. Verkettungen von AWS-Objekten und die logische Verknüpfung von API-Operationen sind notwendig geworden. Der eigentliche Soll-Ist-Vergleich, als Prüfung eines erwünschten Konfigurationszustands (Soll) und einer aktuelle Cloud-Konfiguration (Ist), kommt weiterhin mit relativ einfachen Regelsätzen aus.

Objektspeicher

Der Objektspeicherdienst bei AWS nennt sich Simple Storage Service (S3) und kann jegliche Art von unstrukturierten Daten in nahezu unbegrenzter Menge speichern. Die Speicherung der Objekte erfolgt in einer flachen Struktur innerhalb sogenannter *Buckets*.

Die S3 Buckets können ebenfalls über eine Programmierschnittstelle verwaltet werden. Es werden zwei unterschiedliche APIs für den Objektspeicher angeboten, sowie eine zusätzliche API für AWS Outpost.

Der `s3` API-Satz bietet vor allem Operationen auf Ebene der Buckets und der Objekte, wohingegen das `s3 Control` API für Operationen auf Ebene des Kunden-Accounts dient. Eine exakte Funktionstrennung besteht jedoch nicht.

Das `s3` API weicht etwas von der typischen Namenskonvention der sonstigen AWS API-Operationen ab. Anstatt der typischen Describe<Service> Operationen werden derzeit 31 Get<Service> Aufrufe unterstützt, sowie mehrere Create-, Delete-, Put-, Select- und List-Operationen.

Die Untersuchung des APIs des Objektspeichers soll analog zu den vorherigen Analysen durch eine Auswahl an Leitfragen geführt werden.

Angewendete sicherheitsrelevante Leitfrage(n) für die API-Analyse:

1. Wie kann erkannt werden, ob die Daten im Objektspeicher im Ruhezustand verschlüsselt werden?
2. Werden die Daten auf dem Transportweg zum Objektspeicher verschlüsselt?
3. In welcher geographischen Region sind die Daten des Objektspeichers?
4. Besteht ein Zugriffsschutz für die Daten, oder sind diese ohne Authorisierung zugänglich?

5. Sind die Buckets und Objekte öffentlich zugänglich, oder sind diese nur aus einem privaten Netz erreichbar?
6. Werden zum Schutz Access Control List (ACL) eingesetzt, oder erfolgt der Zugriffsschutz über Richtlinien (Policies) und Identitätsmanagement? AWS empfiehlt Access Control List (ACL) zu vermeiden und zu blockieren.

Bevor auf die Erkenntnisse aus der API Analyse eingegangen werden kann, sind einige wenige Grundlagen hinsichtlich der AWS Sicherheitsmechanismen anhand der gestellten Leitfragen zu erläutern. Die notwendigen Grundlagen werden in der Abfolge der oben aufgeführten Leitfragen erarbeitet.

Bei der Verschlüsselung der Daten im Ruhezustand (englisch: data at rest) sind grundsätzlich zwei Varianten zu unterscheiden. Die *server-seitige*, sowie die *client-seitige* Verschlüsselung der Objekte. Bei der server-seitigen Variante findet die Verschlüsselung erst nach dem Datentransport auf dem Server statt. Die Daten werden im Klartext übertragen und idealerweise durch eine Transportverschlüsselung abgesichert. Bei der client-seitigen Verschlüsselung werden die Daten auf dem sendenden Client bereits verschlüsselt und erst danach an den Server gesendet. Beide Varianten werden bei AWS unterstützt. Das AWS Encryption SDK kann für die client-seitige Verschlüsselung eingesetzt werden. Dieser Ansatz der Verschlüsselung kann über das API jedoch nicht nachgeprüft werden. Dazu müssten die Objekte direkt untersucht werden, beispielsweise über eine Berechnung der Entropie. Daher wird die Variante der client-seitigen Verschlüsselung im Rahmen dieser Arbeit nicht weiter betrachtet.

Die Verschlüsselung ruhender Daten erfolgt laut aktueller Dokumentation mit dem 256-bit Advanced Encryption Standard (AES-256) GCM, wenn sogenannte Amazon S3 verwaltete Schlüssel genutzt werden. Werden andere Algorithmen benötigt, ist auf eine client-seitige Verschlüsselung zurückzugreifen.

Amazon S3 erlaubt sowohl unverschlüsselten Zugriff per HTTP, als auch über Transportverschlüsselung (SSL/TLS) gesicherte Datenkommunikation. Um eine unverschlüsselte Kommunikation mit S3 Buckets zu unterbinden, ist es notwendig eine entsprechende AWS Richtlinie (Policy) zu erstellen und dem Objektspeicher zuzuweisen. Dabei ist die entscheidende Direktive `aws:SecureTransport`, die als Bedingung für den S3-Zugriff geprüft wird. Dies kann auf einzelne Buckets, als benannte Ressourcen, limitiert werden.

Das Listing 4.5 zeigt eine entsprechende Policy, die jeglichen lesenden oder schreibenden Zugriff (Zeile 6) auf ein definiertes S3 Bucket (Zeilen 8-9) nur mittels Trans-

portverschlüsselung (Zeile 13) erlaubt, ansonsten die Anfrage ablehnt (Zeile 4). Diese Richtlinie gilt für alle Identitäten und Berechtigungsrollen (Zeile 5).

```
1  {
2     "Statement": [
3        {
4           "Effect": "Deny",
5           "Principal": "*",
6           "Action": "s3:*",
7           "Resource": [
8              "arn:aws:s3:::ch-gollwitzer-mt-bucket01",
9              "arn:aws:s3:::ch-gollwitzer-mt-bucket01/*"
10          ],
11          "Condition": {
12             "Bool": {
13                "aws:SecureTransport": "false"
14             }
15          }
16       }
17    ]
18 }
```

Listing 4.5: Bucket Policy die den Zugriff nur per SSL/TLS erlaubt

Zur Analyse der Frage 4, bezüglich der öffentlichen Zugriffsmöglichkeit auf einen Objektspeicher, ist es wichtig den folgenden Grundsatz von AWS zu verstehen: Amazon S3 betrachtet eine Bucket- oder Objekt-ACL als öffentlich, wenn sie Mitgliedern der vordefinierten Gruppen *AllUsers* oder *AuthenticatedUsers* irgendwelche Berechtigungen erteilt. AWS geht bei der Prüfung der Zugriffskontrollen zunächst von der Annahme aus, dass die Richtlinie öffentlich ist. Er wenn die Richtlinie nicht-öffentliche Regeln vorgibt, wird die Konfiguration als nicht-öffentlich eingestuft.

Ein zweiter Indikator, ob ein Zugriffsschutz auf ein Bucket beziehungsweise dessen Objekte vorhanden ist, kann über die Prüfung der Bucket-Policies gegeben sein (Frage 6). Existiert eine Bucket-Policy die eine Regel beinhaltet die eine Aktion (`Action` Direktive) zulässt (`Effect` Direktive mit Wert 'Allow'), ohne dass die Erlaubnis auf einen bestimmten Benutzer, eine Rolle oder Gruppe eingeschränkt wird (`Principal` Direktive mit Wert '*'), wird keine Autorisierungsprüfung durchgeführt. Wäre im Listing 4.5 die Direktive `Effect` auf `Allow` gesetzt, wäre dies eine problematische Konfiguration. Es sind somit sowohl die ACL, als auch die Bucket-Policies auf Konfigurationsprobleme zu untersuchen.

Mit dem bereits erarbeiteten Wissen um die Prinzipen der S3 Bucket-Policies (siehe Listing 4.5), lässt sich auch eine Möglichkeit des Zugriffsschutzes des Objektspeichers anhand der Netzwerkadressierung (OSI Schicht 3) herstellen, wie dies in Leitfrage 5 angefragt ist. Der Block der Bedingungen (englisch Condition) ermöglicht es auf

die Quelladresse des anfragenden Client zu untersuchen und nur bestimmte CIDR-Bereiche zuzulassen. Wie in Listing 4.6 dargestellt, wird die Prüfung `idAddress` (Zeile 2) auf den Inhalt des Parameters `aws:SourceIp` (Zeile 3) angewendet. Nur der angegebene IP-Adressraum kann nun auf die Ressourcen zugreifen (Zeile 4). Als Quelle könnte auch eine AWS Virtual Private Cloud (VPC) selektiert werden, wozu jedoch weitere AWS Cloud-Elemente eingesetzt werden müssten.

```
1 "Condition": {
2   "IpAddress": {
3     "aws:SourceIp": [
4       "93.175.119.0/24"   # IP aus Adressraum von WINGS
5     ]
6   }
7 }
```

Listing 4.6: Bucket Policy mit Bedingung auf die Quell-IP des Clients

Auf Basis der oben erarbeiteten Erkenntnisse wurden ausgewählte API-Operationen des Objektspeicherdienstes identifiziert und untersucht. Die relevanten Schlüssel und Werte zur automatisierten Prüfung und Absicherung der Cloud-Konfiguration wurden erarbeitet und in Tabelle 13 zusammengefasst.

Erkenntnisse Die in Tabelle 13 (siehe Anhang A.3.8) zusammengefassten Ergebnisse der API-Analyse bezüglich des Objektspeichers zeigen umfangreiche Möglichkeiten die Sicherheitseinstellungen der S3 Buckets und der Objekte vornehmen zu können. Die Prüfung der Bucket-Policies, als auch der ACL, spielen eine entscheidende Rolle. Die Konfigurationseinstellungen der Bucket-Policies sind sehr umfangreich und können hier nur im Ausschnitt aufgezeigt werden. Es können einige grundlegende Sicherheitsmängel erkannt werden. Viele der Konditionen zur Autorisierung werden aber in der Realität sehr spezifisch für das jeweilige Unternehmen oder die Organisation sein, um die Berechtigungen fachlich präzise einzustellen. Bei der Interpretation der aktuellen Konfiguration ist aber auch Vorsicht geboten. Beispielsweise kann eine aktuell konfigurierte Standard-Verschlüsselung (englisch default) aktiviert sein und man könnte annehmen, dass nun alle Objekte auch verschlüsselt sind. Die Standard-Verschlüsselung bezieht sich aber stets nur auf *hinzukommende* Objekte. Das bedeutet, sollte die Standard-Einstellung erst später gesetzt worden sein, können teilweise unverschlüsselte Objekte im Bucket vorhanden sein.

4.2.4 Virtualisiertes Netzwerk

In den vorangegangenen Kapiteln wurden bereits erste AWS-Dienste aus dem Bereich der virtualisierten Netzwerke und auch einzelne Netzwerk-Sicherheitsdienste genannt. Diese sollen in diesem Kapitel näher betrachtet werden, deren Zusammenwirken aufgezeigt werden und die Fähigkeiten des API in diesem Kontext analysiert werden. Die API-Analyse wird sich dabei auf einige wesentliche und ausgewählte AWS-Netzwerkdienste konzentrieren. Die Analyse des API soll auch hier durch eine getroffene Auswahl an Fragestellungen geführt werden.

Angewendete sicherheitsrelevante Leitfrage(n) für die API-Analyse:

1. Wie kann der Zugriff auf Netzwerke eingeschränkt und kontrolliert werden?
2. Wie erfolgt der Zugriff in und aus dem Internet auf die Netze und Ressourcen, wie kann dies abgesichert werden?
3. Kann eine Absicherung gegen Denial of Service Attacken geprüft werden?
4. Lassen sich getroffene Maßnahmen gegen Angriffe von geteilten Netzdiensten, wie dem Domain-Name Dienst, in der Cloud-Konfiguration erkennen?
5. Sind Firewall-Dienste und deren Regelkonfiguration über das API steuerbar?

Die Virtual Private Cloud

Die Virtual Private Cloud (VPC) stellt ein isoliertes virtualisiertes Netzwerksegment in der AWS-Infrastruktur dar, die eindeutig einem AWS Kunden zugeordnet ist. Jeder Mandant kann theoretisch beliebig viele VPCs verwalten. Die VPC definiert einen privaten IP-Adressraum durch einen CIDR-Block (IPv4 und IPv6) wie in RFC 1918 definiert. Die VPC existiert in exakt einer AWS-Region. Soll die Cloud-Umgebung auf mehrere AWS-Regionen aufgeteilt sein, sind mehrere VPCs anzulegen. Die VPC kann in *Subnetze* untergliedert werden. Je Availability Zone (AZ) in einer Region sind eigene Subnetze anzulegen. Das Subnetz stellt dabei einen IP-Teilbereich der VPC dar. Das Bild 12 veranschaulicht diese Zusammenhänge. *Routing-Tabellen* und deren Routen steuern den Netzwerkverkehr von und zu den Subnetzen.

Ist einem Subnetz eine Routing-Tabelle zugewiesen, die über eine Route zu einem Internet-Gateway verfügt, wird dieses als *öffentliches* Subnetz bezeichnet.

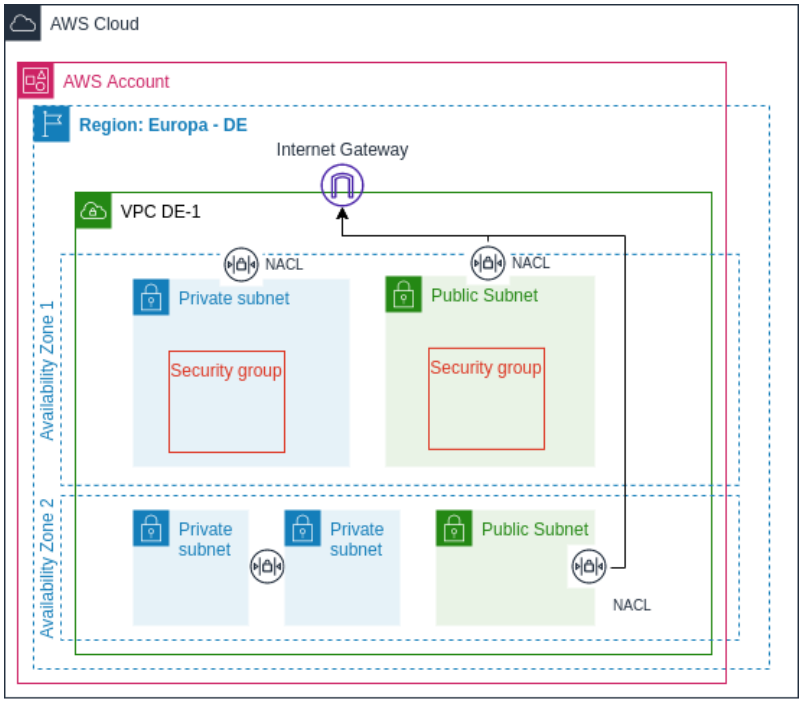

Bild 12: Genereller Aufbau AWS Netzwerk aus VPC und Subnetzen - mit einem Internet-Gateway

Zugriffsteuerung auf Netzwerkebene

Die Kommunikation zwischen Teilnetzen erfolgt auf Basis der Routing-Tabellen und kann durch *Netzwerk-Zugriffssteuerungslisten* (englisch Network Access Control List (NACL)) kontrolliert und gesichert werden. Eine NACL erlaubt oder verweigert definierten ein- oder ausgehenden Datenverkehr auf Ebene der Subnetze. Jede VPC verfügt über eine Standardnetzwerk-ACL die jeglichen ein- und ausgehenden Datenverkehr (IPv4 und IPv6) erlaubt. Eine Netzwerk-ACL hat Regeln für eingehenden und ausgehenden Datenverkehr. Jede Regel kann Datenverkehr entweder erlauben oder verweigern. Zu beachten ist, dass Netzwerk-ACLs zustandslos sind, d.h. Antworten auf zulässigen eingehenden Datenverkehr unterliegen den Regeln für ausgehenden Datenverkehr und umgekehrt. Fachlich besteht eine Regel aus dem Protokoll (Standard Internet-Protokollnummern nach IANA), dem Port-Bereich, der Quelle oder dem Ziel (ein- bzw. ausgehend) als CIDR-Bereich, sowie der Aktion: erlauben oder verweigern.

Absicherung des Netzwerkzugriffs einzelner Ressourcen

Im Gegensatz dazu wirkt die Sicherheitsgruppe auf der Ebene einer einzelnen AWS-Ressource. Dies wurde bereits im Kontext der Absicherung des Elastic File System (EFS) aufgezeigt. Die Security Group (SG) wirkt dabei wie eine Host-Firewall, wird jedoch nicht auf dem Host installiert. Eine SG-Konfiguration kann zum Schutz einer EC2-Instanz oder Datenbank eingesetzt werden, um den ein- und ausgehenden Datenverkehr zu dieser einen Ressource zu kontrollieren. Die Sicherheitsgruppe arbeitet wie eine klassischer zustandsbehafteter Paketfilter auf Basis von Regeln, die sich aus der Richtung (ein-/ausgehend), dem Protokoll (TCP/UDP/ICMP), dem Portbereich, sowie Angaben zu Quelle und Ziel auf Basis der CIDR-Notation zusammensetzen.

Internetzugriff steuern und sichern

Das Internet Gateway (IGW) ist eine VPC-Komponente die die Kommunikation zwischen VPC und dem Internet ermöglicht. Öffentlich erreichbare Dienste sind zunehmend Angriffen auf Basis sogenannter Distributed Denial of Service (DDoS) Techniken (MITRE TT1499 bzw. T1464) ausgesetzt [31]. Dabei wird die Kapazität der Netzwerkbandbreite, oder der Dienst-Endpunkte angegriffen und diese durch massenhafte Anfragen überlastet. Spezialisierte Dienstanbieter bieten Schutzmaßnahmen an, die den eigenen Diensten vorgeschaltet werden können. Auch AWS bietet DDoS-Protektion als einen verwalteten Dienst unter dem Produktnamen *AWS Shield* an. AWS Shield ist dabei eine Funktion des Web Application Firewall (WAF) Dienstes von AWS.

Der Dienst AWS Shield Advanced wird leider nicht im Rahmen dieser Thesis betrachtet, da die Kosten mit $3.000USD pro Monat zuzüglich der Datentransferkosten und einer Mindestlaufzeit des Abonnement von 1 Jahr durchaus erheblich sind. Der kostenlose Dienst AWS Shield Standard ist permanent aktiviert und kann auch nicht parametrisiert werden. Eine Betrachtung im Rahmen der Cloud Security Posture Management (CSPM) Automatisierung ist daher nicht notwendig.

Die Kernfunktion der AWS Web Application Firewall (WAF), der Schutz von Web-Anwendungen, sowie deren Konfiguration lässt über das API analysieren und steuern. Die Konfigurations zum Schutz der Informationssicherheit der Web Application Firewall (WAF) wird über Web-Zugriffskontrollliste (englisch web access control list (Web-ACL)) gesteuert. Eine Web-ACL prüft dabei die HTTP(S) Kommunikation

zwischen dem Client und den AWS Ressourcen und schützt beispielsweise vor Angriffen wie SQL Injektion oder Skript-Angriffen wie Cross-Site-Scripting (XSS). Über das Query-API können Web-Zugriffskontrolllisten erzeugt (Action: `CreateWebACL`), ausgelesen (Action: `GetWebACL`) und auch gelöscht werden (Action: `DeleteWebACL`). Die Syntax der Web-ACL nutzt die JSON-Notation und könnte automatisiert analysiert werden. Dabei setzten die Web-ACL Regeln auf umfangreiche Möglichkeiten die HTTP(S) Kopf- und Nutzdaten zu analysieren, u.a. durch Einsatz regulärer Ausdrücke, und entscheiden darauf den Datenverkehr zuzulassen, zu überwachen oder sogar zu blockieren. Die WAF schützt dabei nicht die einzelnen Web-Server auf EC2-Instanzen, sondern vorgelagerte AWS Dienste wie die Lastverteiler (Application Load Balancer) oder CloudFront Dienste zur global skalierten Web-Contentverteilung.

Absicherung des DNS Route53

Route53 ist ein skalierbarer Domain Name System (DNS)-Dienst von AWS. Der Dienst kann über die Query API gesteuert und verwaltet werden. Route53 kann über das API konfiguriert werden, um das transportverschlüsselte DNS-Protokoll Domain Name System Security Extensions (DNSSEC) zu aktivieren, um die eigene Domäne gegen DNS-Angriffe mittels DNS-Spoofing oder Man-In-the-Middle-Angriff zu schützen.

Der Datenschutz des Route53 bei WHOIS-Abfragen kann auch über das API aktiviert und deaktiviert werden. Standardmäßig wird die Datenschutzfunktion bereits für registrierte Domänen aktiviert, sodass die Kontaktinformationen nicht übermittelt werden.

DNS-Konfigurationen, die dem Spam- und Spoofing-Schutz dienen können, wie die Aktivierung des Sender Policy Framework (SPF), können ebenfalls mittels der AWS-API und der Domänen-Einträge gesteuert werden.

Das API des Route53 ermöglicht auch die Zugriffs- und Berechtigungsverwaltung der administrativen Funktionen des DNS-Dienstes. Hier kommen wiederum Richtlinien (Policy) und das AWS IAM zum Einsatz. Dieser Bereich ist sehr umfangreich und kann sehr präzise gesteuert werden. Amazon Route53 verfügt über 75 einzelne administrative Aktionen die in den Richtlinien erlaubt oder verhindert werden könnten. Diese Richtlinien könnten über einen automatisieren CSPM-Ansatz ausgelesen und geprüft werden, sind sehr wahrscheinlich in Realität jedoch äußerst unternehmensspezifisch ausgeprägt. Eine detaillierte API-basierte Sicherheitsprüfung der Route53

Zugriffs- und Berechtigungsverwaltungerscheint jedoch nicht zielführend und wird im Rahmen dieser Thesis daher nicht weiter ausgeführt.

Verschlüsselte Datenverbindung

Soll der Zugriff auf die AWS-Ressourcen in der virtuellen Cloud nicht ungeschützt über das Internet erfolgen, oder sollen die Ressourcen nur für einen eingeschränkten Nutzerkreis zugänglich sein, bieten sich auch der Einsatz virtueller private Netzwerke (VPN) an. Der VPN-Dienst von AWS unterstützt sowohl site-to-site VPN, also auch Client-VPN Verbindungen. Site-to-site VPN werden gerne genutzt, wenn beispielsweise eine Niederlassung oder ein Rechenzentrum relativ kostengünstig mit der AWS-Cloudumgebung gesichert verbunden werden sollen. Client-VPN Verbindungen können eine gesicherte Verbindung zu Diensten in der AWS-Cloud für mobile Mitarbeiter bieten.

Der AWS VPN-Dienst unterstützt verschiedene Sicherheitsmaßnahmen bei der Authentifizierung, wie die Authentifizierung über Active Directory, mittels Zertifikaten und auch über föderierte Authentifizierung über SAML-2.0. Zusätzlich wird die Multi-Factor Authentication (MFA) angeboten. Auch werden Sicherheitsfunktionen, wie eine Certificate Revocation Lists (CRL) - zur Schwarzlistung von Zertifikatssperrungen - ermöglicht. Durch eine Kombination aus Subnetz, Client-VPN-Endpunkt und den Sicherheitsgruppen können Zugriffe auf AWS-Ressourcen auf der Netzwerkebene kontrolliert und eingeschränkt werden.

API Zusammenfassung und Erkenntnisse

Die erarbeiteten Ergebnisse zu den Fähigkeiten des API im Bereich der virtualisierten Netzwerke und der Netzwerkabsicherung sind in Tabelle 14 (siehe Anhang A.3.9) zusammengefasst. Das API ist im Bereich des Netzwerks sehr umfangreich und bietet bisher keine erkennbaren Einschränkungen sicherheitsrelevante Informationen automatisiert zu ermitteln. Es ist auch hier erkennbar, dass automatisierte Prüfregeln mit relativ einfachen Vergleichslogiken auskommen können. Die logische Vernetzung der einzelnen AWS-Objekte, über deren Primärschlüssel, ist eine wesentliche benötigte Fähigkeit der Regellogik. Anzumerken ist, dass die in diesem Abschnitt erarbeiteten API-Operationen und Schlüssel-Wertpaare nur einen Ausschnitt darstellen können.

4.2.5 Schlüsselverwaltung

Für die Verwaltung kryptografischer Schlüssel bietet AWS den Key Management Service (KMS). Es können sowohl symmetrische, wie auch asymmetrische Schlüssel verwaltet werden. Diese verwalteten Schlüssel können erzeugt, deaktiviert und gelöscht werden. Die Schlüssel können in verschieden Anwendungsbereichen, beispielsweise zur Verschlüsselung von Datenspeichern oder zur Authentifizierung, genutzt werden.

Um über das API Operationen zur Verwaltung der Schlüssel ausführen zu können, sind spezielle Berechtigungen, z.B. `kms:createKey`, und darauf aufbauend passende Policies zu erstellen.

Angewendete sicherheitsrelevante Leitfrage(n) für die API-Analyse:

1. Kann die Funktion des Schlüssels ermittelt werden und ob dieser aktiv ist?
2. Können die unterstützten kryptographischen Verfahren / Algorithmen ermittelt werden?
3. Kann der Schlüssel deaktiviert werden?
4. Kann erkannt werden, ob der Schlüssel einem definierten Rotationsverfahren unterliegt?
5. Kann erkannt werden, ob ein spezielles Hardware-Sicherheitsmodul (HSM) genutzt wird?

Erkenntnisse Die API-Actions unterstützen die grundlegenden Funktionen zur Schlüsselanalyse, jedoch werden oftmals nur generische Werte zurückgegeben. Ohne weitere Dokumentation beziehungsweise aktuelle Kenntnis der Standards des CSP sind die Rückgabewerte nicht nützlich. Speziell im Bereich der Analyse der Algorithmen und Verfahren sind diese zu oberflächlich. Die Ergebnisse der API-Analyse sind in Tabelle 15 zusammengefasst.

4.3 Analyse Infrastructure-as-Code Ansatz

4.3.1 Auswirkung auf den bisherigen CSPM-Ansatz

In Kapitel 3.1 wurde ein statisches Model für den Cloud Security Posture Management (CSPM) Ansatz aufgebaut. Das bisherige Modell geht dabei von der API-basierten Analyse der Cloud-Ressourcen aus. Eine Sicherheitsauditierung kann somit nur gegen bereits konfigurierte, oder sogar bereits aktiv laufende Cloud-Ressourcen erfolgen, da die API des Cloud-Anbieters nur Informationen zu *produktiven* Cloud-Ressourcen bereitstellen kann. Cloud-Ressourcen die sich noch in einem Planungszustand, oder im Rahmen einer Entwicklungstätigkeit der Infrastruktur befinden, können damit nicht auditiert werden.

Die Entwicklung von Cloud-Infrastrukturen und -Konfigurationen auf Basis eines softwaregestützten Verfahrens nutzt den sogenannten Infrastructure as Code (IaC)-Ansatz. Die Cloud-Infrastruktur wird dabei in Form von Konfigurationsdateien virtuell entwickelt und definiert. Die Cloud-Dienste und deren Konfigurationszustand werden auf Basis einer speziellen Sprache und Syntax definiert. Diese Konfiguration, oftmals in Form von Dateien, wird danach durch einen automatisierten Installationsvorgang in der Cloud-Umgebung eines Cloud Service Provider (CSP) verfügbar gemacht.

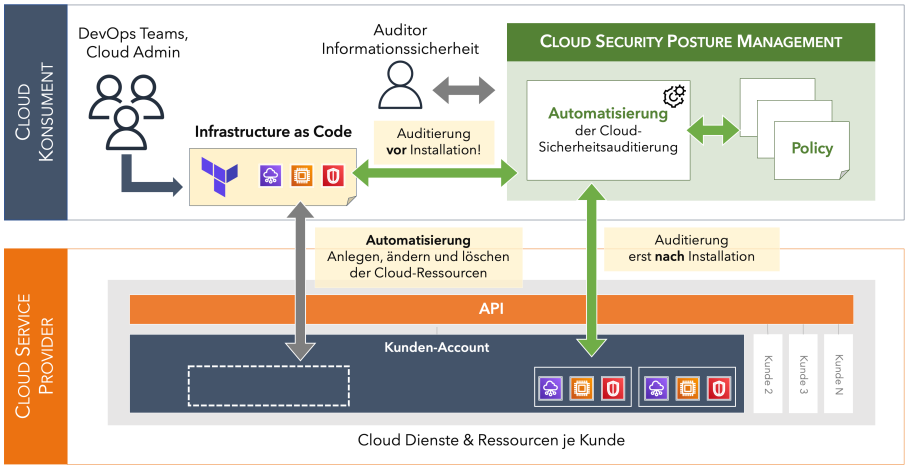

Bild 13: Schematische Darstellung CSPM - Ausbaustufe mit IaC

Bild 13 zeigt diesen strukturellen Unterschied auf. Die DevOps-Teams, bzw. Cloud-Administratoren, erstellen die IaC-Konfigurationen außerhalb der eigentlichen Cloudumgebung. Diese Cloud-Konfiguration ist somit noch nicht produktiv in der Cloud-

Umgebung installiert. Ein Cloud Security Posture Management (CSPM)-Werkzeug prüft nun gegen diese IaC-Konfiguration Dateien und kann Sicherheitsmängel somit bereits vor der Installation in der Cloud-Umgebung erkennen und potentiell verhindern.

4.3.2 Analyse aufbauend auf den API-Erkenntnissen

Das Produkt *Terraform (TF)* der Firma HashiCorp Inc. gilt derzeit als quasi Marktführer in im Segement der Infrastructure as Code (IaC) Anbieter. Zur Definition der Cloud-Ressourcen und deren Konfiguration bietet Terraform eine eigene Konfigurationssprache namens HashiCorp Configuration Language (HCL) an. Die Sprache besteht aus Elementen die nicht abhängig sind von einer Cloud-Zielumgebung, sowie spezifischen Sprachelementen die nur für eine konkrete Cloud-Zielumgebung nutzbar sind. Eine TF-Konfiguration ist somit immer auf genau eine konkrete Cloud-Zielumgebung, d.h. für einen konkreten CSP, anwendbar. Eine direkte Nutzung der TF-Konfiguration in einer anderen CSP-Infrastruktur ist nicht möglich.

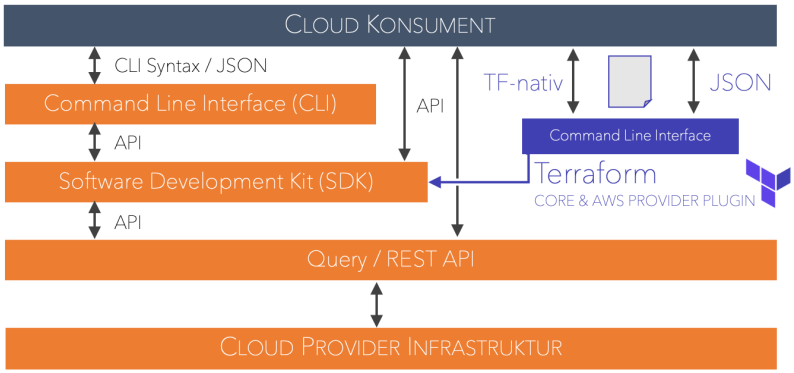

Bild 14: Gegenüberstellung AWS Programmierschnittstellen und Terraform API / CLI

In diesem Kapitel wird die Syntax und die wesentliche Sprachelemente von HCL aufgezeigt. Eine umfängliche Analyse, wie diese im vorangegangenen Kapitel 4.2 stattgefunden hat, ist nicht erneut notwendig. Terraform baut prinzipiell auf dem API des jeweiligen Cloud Service Provider (CSP) auf, wodurch die Benennungen der Cloud-Ressourcen und deren Attribut-Wert-Paare sehr ähnlich zu denen des CSP und dessen API sind. Die erarbeiteten Ergebnisse aus der API-Analyse lassen sich auf die Terraform-Sprache übertragen.

Das Bild 14 zeigt in einer Gegenüberstellung die unterschiedlichen technischen Zugriffsebenen des Cloud-Anbieters, sowie die Interaktion zwischen Nutzer und Cloud-

Infrastruktur im Falle des Produkts Terraform. Terraform bietet ein Kommanodzeilenwerkzeug das zwei unterschiedliche Datenformate verarbeiten kann. Einerseits das HashiCorp native Datenformat HashiCorp Configuration Language (HCL), sowie die zweite Option auf Basis des Standards JavaScript Object Notation (JSON). Das native Datenformat wird typischerweise bevorzugt, da es für den Menschen lesbarer ist und auch Werkzeuge zum Design der Infrastruktur dieses Format bereitstellen.

4.3.3 Sprachaufbau Terraform HCL

Das Kernelement in der HCL stellen die Ressourcen (Schlüsselwort `resource`) dar, die ein oder mehrere Infrastruktur-Objekte repräsentieren. Ressourcen sind beispielsweise virtuelle Maschinen, oder virtualisierte Netzwerke oder Netzwerk-Komponenten, können aber auch detaillierte Komponenten wie DNS-Einträge darstellen.

Die drei wesentlichen Grundelemente der Sprache sind:

- *Blöcke* sind gruppierende Strukturen die andere Konfigurations-Objekte, wie z.B. Ressourcen, zusammenfassen. Blöcke werden typisiert über einen block type und können einen oder mehrere Bezeichner mittels block label tragen. Der body stellt den eigentlichen Block-Inhalt mittels sogenannten Argumenten dar. Blöcke können auch geschachtelt werden.
- *Argumente* weisen einen Wert einem Namen zu. Argumente existieren in Blöcken.
- *Ausdrücke* stellen einen Wert dar. Entweder direkt, oder über eine Referenzierung oder als Kombinationen. Sie werden in Argumenten oder auch anderen Ausdrücken genutzt.

Die Ressource wird über einen Ressourcen Typ konkret für einen Terraform-Provider ausgeprägt. Ein *Provider* ist z.B. AWS, GCP oder Azure. Eine Ressource, ein Block, kann einen sprechenden Namen erhalten über den sogenannten Block Label. Für jede Ressource werden spezifische Konfigurationsargumente, wie z.B. einen Identifierer, oder eine Provider-spezifische Angabe zur virtualisierten Kapazitätsgrößen, angeben. Die möglichen Konfigurationsargumente werden über den Ressourcen-Typ festgelegt.

Das Listing 4.7 zeigt ein lauffähiges Grundgerüst einer Terraform-Konfigurationsdatei.

Beispiel

```
1  # Kopfdaten für eine TF-Konfiguration
2  terraform {
3    required_providers {
4      aws = {
5        source = "hashicorp/aws"
6        version = "-> 3.27"
7      }
8    }
9  }
10
11 # Den Provider, also den CSP, festlegen: AWS
12 provider "aws" {
13   profile = "default"
14   region  = "eu-central-1"
15 }
16
17 # Erzeugen einer AWS EC2-Instanz
18 resource "aws_instance" "MT-Gollwitzer_EC2-Demo" {
19   ami = "ami-0a634ae95e11c6f91"
20   instance_type = "t2.micro"
21 }
```

Listing 4.7: Beispiel einer Terraform-Vorlage im HCL-Format

Der erste Block (Zeile 2-9), beginnend mit dem block type-Schlüsselwort `terraform`, stellt die Kopfdaten bereit, definiert welcher Provider genutzt wird und gibt eine Versionsangabe zur genutzten HashiCorp-Terraform Sprache an. Der zweite Block `provider` (Zeile 12-15) konfiguriert bereits übergreifende Parameter, wie die AWS-Region in der die Ressourcen installiert werden sollen. Dies ist bereits eine Provider-spezifische Angabe, da der Resourcen-Typ 'aws' deklariert wird. Den Regionen-Code in Zeile 14 kennen wir bereits aus der API-Analyse aus Kapitel 4.2.1. Der dritte Block konfiguriert eine EC2-Instanz und verwendet dazu den Ressourcen-Typ *aws_instance* (Zeile 18-21). Die genutzte Bezeichnung der Ressource 'MT-Gollwitzer_EC2-Demo', das sogenannte block label, kann frei gewählt werden.

Die möglichen Block-Typen, deren Argumente und Werte, sind Provider-spezifisch und werden bei HashiCorp in sogenannten Registries dokumentiert. Wie in Listing 4.7 bereits erkennbar, tragen die jeweiligen Schlüsselworte in der Regel einen Provider-abhängigen Präfix: *aws_*<Ressource>.

In dem einfachen Beispiel lässt sich auch die starke und gewollte Ähnlichkeit der Benennungen erkennen. Eine virtuelle Maschine wird in dem AWS-API als *Instance* bezeichnet (siehe Kapitel 4.2.2). Der Provider AWS bei Terraform nutzt ebenfalls den Bezeichner `instance` in Verbindung mit dem Präfix `aws_`. Das der virtuellen Maschine zugrundeliegende Abbild (der Snapshot) wird mit der Abkürzung für das Amazon Machine Image (AMI) deklariert (Zeile 19). Auch die Definition, welche

Hardware-Klasse verwendet werden soll, basiert auf der AWS Klassifikation in Form der Instanz-Typen (englisch instance type) und wird in TF übernommen durch das Argument `instance_type` (Zeile 20). Auch der Wert des Arguments `t2.micro` wird exakt wie bei dem AWS API angegeben.

Die Verwaltung von Cloud-Ressourcen mit dem Terraform Produkt folgt einem definierten Lebenszyklus. Die Lebensyzklus-Zustände, sowie deren Übergang ist im Anhang A.5.1 dokumentiert.

Für die CSPM-Analyse wird im Rahmen der prototypischen Umsetzung, siehe Kapitel 6, auf Basis des sogenannten Terraform *Plan* gearbeitet werden. Dieser Zustand wird von Terraform erstellt, bevor Änderungen in die Cloud-Umgebung übertragen werden. Dieser Zustand ist somit vor dem eigentlichen Deployment und bietet die Möglichkeit Sicherheitsmängel frühzeitig zu erkennen und zu beheben. Ein weiterer Vorteil ist, dass dieser Plan in ein gängiges Format (JSON) konvertiert werden kann.

4.4 Zusammenfassung

Die Analyse des AWS Cloud-API und die daraus gewonnen Erkenntnisse über die Fähigkeiten und Einschränkungen bilden die Basis für darauf aufbauende Technologien wie Terraform. Ein Großteil der gestellten Sicherheitsleitfragen ließ sich erarbeiten und konnte mit den identifizierten API-Operationen und Attributen gelöst werden. Es mussten aber auch Einschränkungen festgestellt werden, die je Cloud-Ressource spezifisch sind. Die Möglichkeiten über die Cloud-API eine automatisierte Sicherheitsprüfung zu erarbeiten überwiegen aber deutlich die erkannten Einschränkungen des API.

In dem nun folgenden Kapitel werden bewährte Methoden (englisch: best-practises) erhoben und bewertet, die als mögliche Grundlage oder Muster für eine prototypische Umsetzung dienen könnten.

5 Erhebung und Bewertung bewährter Methoden

5.1 Methodik

Der Cloud Security Posture Management (CSPM) Ansatz versucht Sicherheitsmängel in der Konfiguration von Cloud-Ressourcen, die durch den Cloud-Konsumenten verursacht werden, automatisiert zu erkennen. Die Sicherheitsmängel sollen durch eine Art Soll-Ist-Vergleich gegen bewährte Methoden (englisch: Best-Practises) oder unternehmenseigene Sicherheitsvorgaben detektiert werden. Der Soll-Zustand stellt dabei die Sicherheitsvorgaben dar, der Ist-Zustand die aktuelle Konfiguration der Cloud-Infrastruktur. Da der Vergleich automatisiert zu erfolgen hat, müssen sich die Sicherheitsregeln der Best-Practises in maschinenlesbare Regeln transformieren lassen oder sogar bereits als Pseudocode vorliegen.

In diesem Kapitel wird eine getroffene Auswahl an öffentlich verfügbaren Quellen verschiedener Sicherheitsinstitutionen, aus unterschiedlichen Bereichen und geografischen Regionen bezüglich verfügbarer Best-Practises untersucht. Die veröffentlichten bewährte Methoden werden geprüft, inwieweit diese als Grundlage zur Automatisierung der IaaS-Informationssicherheit dienen können. Die Quellen umfassen Cloud-Anbieter, Behörden oder Verbände sowie kommerzielle und gemeinnützige Organisationen und Unternehmen. Die Auswahl der kostenfrei und öffentlich verfügbaren Quellen erfolgte nach Relevanz zu Informationssicherheit und Bezug zu Cloud-Infrastruktur, nationaler und internationaler Anerkennung oder Bekanntheitsgrad sowie der Sprache (Deutsch oder Englisch).

Die Quellen werden untersucht nach deren Umfang in den für IaaS relevanten Funktionsbereichen (siehe Tabelle 1 in Kapitel 4.1), dem Grad der Nutzbarkeit für die CSPM-Automatisierung und der Aktualität der Informationen.

Zielsetzung ist es, eine relevante Marktübersicht zu erhalten sowie die Quellen und Anbieter in den genannten Feldern zu bewerten. Dies dient als Basis für die prototypische Umsetzung im Rahmen dieser Arbeit und kann für einen späteren Praxiseinsatz genutzt werden.

5.2 Best-Practises des Cloud Providers

Der Dokumentationsumfang bei AWS kann generell als sehr umfangreich bewertet werden. Aus der Fülle an Informationen wurden zunächst die als wesentlich erscheinenden Dokumente mit Empfehlungen zur Informationssicherheit herausgearbeitet. Als relevante Quellen für eine Analyse wurden folgende Dokumente identifiziert:

- Der Säule *Sicherheit* aus dem AWS Well-Architected Framework [38].
- Die AWS Security Reference Architecture (SRA) [39].
- Die AWS Security Dokumentation [40].

Ergänzend sei hier noch die Sammlung von White-Papern, Tutorials, Blogs und Handbüchern aus der Internetseite *Bewährte Methoden für Sicherheit, Identität und Compliance* [41] erwähnt, die als weiterer Fundus dienen könnten. Diese Quelle wird aber im Rahmen dieser Thesis nicht untersucht.

Für die thematische Einführung in die AWS-Sicherheitsgrundlagen sind auch die Lerninhalte wie der *Security Learning Plan* im AWS Skill Builder sowie die Literaturempfehlungen zur Personenzertifizierung *AWS Certified Security – Specialty* empfehlenswert.

5.2.1 AWS Well-Architected Framework - Säule Sicherheit

Das Well-Architected Framework beschreibt Prinzipien, Konzepte und bewährte Methoden für die Gestaltung und die Architektur von Cloud-Umgebungen auf Basis der AWS-Infrastruktur. Das Framework ist in sechs Säulen gegliedert. Eine davon behandelt die Informationssicherheit [38].

Der *Security Pillar* ist in einem 77-seitigen Dokument beschrieben, das erstmalig 2016 veröffentlicht und zuletzt im Oktober 2022 überarbeitet wurde. Das Dokument scheint regelmäßig gepflegt zu werden und kann als sehr aktuell eingestuft werden. Die Best-Practises sind in folgende Kategorien gegliedert. (Die Anzahl der Empfehlungen ist in kursiver Schrift dargestellt):

- Generell / *8*
- Identity and access management / *14*
- Detektion (Logging und Events) / *4*

- Netzwerkschutz / *4*
- Schutz der Rechendienste / *6*
- Datenschutz / *13*
- Incident Response / *7*

Die Empfehlungen sind nummeriert, tragen einen eindeutigen Titel und werden jeweils in circa einer halben Seite beschrieben. Je Empfehlung können Umsetzungshinweise (englisch: Implementation Guidance) gegeben sein, die vor allem aus Verweisen auf die AWS-Dokumentation der Cloud-Dienste besteht. Je Best-Practise ist eine Risiko-Einstufung, wenn die Empfehlung nicht umgesetzt wird, dokumentiert. Die Risiko-Stufen sind *Low*, *Medium* und *High*, wobei keine Definition der Stufen existiert.

Die Empfehlungen sind auf unterschiedlichem Abstraktionsniveau verfasst. Beispielsweise behandelt die Empfehlung `SEC01-BP08` *Identify and prioritize risks using a threat model* einen organisatorischen Sicherheitsansatz auf relativ abstrakter Ebene, während `SEC02-BP01` *Use strong sign-in mechanisms* konkrete Anforderungen und Umsetzungshinweise beschreibt. Dazu zählt die Verwendung von Passwort-Richtlinien, Vorgaben zu Passwortlängen, die Unterbindung der Passwortwiederverwendung oder der Einsatz der Multifaktor-Authentifizierung (MFA) mittels einer AWS IAM-Policy.

Umsetzungsbeispiel zur CSPM-Automatisierung Als ein Beispiel, wie auf Basis der Best-Practice (BP) eine automatisierte Sicherheitsregel abstrakt entwickelt werden würde, kann die Best-Practice `SEC08-BP02` *Enforce encryption at rest* dienen.

Die Empfehlung zur Verschlüsselung der Daten im Ruhezustand bezieht sich auf die AWS-Dienste EFS, EBS, dem Amazon Relational Database Service (RDS), S3, sowie auf Amazon Machine Image Abbilder. Es werden auch Empfehlungen für den Einsatz des AWS Encrpytion SDK gegeben, das vor allem für Softwareentwickler relevant ist und in diesem Kontext keinen Nutzen darstellt.

Die BP empfiehlt für S3-Buckets die *default encryption* zu aktivieren. Dies stellt für eine Sicherheitsregel den SOLL-Zustand dar. Der IST-Zustand der S3 buckets, die bereits produktiv sind oder neu erstellt werden, ist dagegen zu prüfen. Der Ansatz soll hier am Beispiel einer Terraform Vorlage dargestellt werden.

```
1  resource "aws_kms_key" "wings-key-s3" {
2    description = "Schlüssel wird für die S3-Buckets Verschlüsselung genutzt."
3  }
4
5  resource "aws_s3_bucket" "mybucket" {
6    bucket = "WINGS Testbucket"
7    server_side_encryption_configuration {
8      rule {
9        apply_server_side_encryption_by_default {
10         kms_master_key_id = aws_kms_key.wings-key-s3.arn
11         sse_algorithm     = "aws:kms"
12       }
13     }
14   }
15 }
```

Listing 5.1: Terraform - S3 Bucket Verschlüsselung

Das Listing 5.1 zeigt wie in einer Terraform Vorlage die serverseitige Verschlüsselung eines S3-Buckets aktiviert wird. In den Zeilen 1-3 wird zunächst ein kryptographischer Schlüssel mit dem AWS Key Management Service (KMS) erzeugt. In den Zeilen 5-15 wird ein S3 Bucket erzeugt. Die Direktive `server_side_encryption_configuration` (Zeile 7) steuert dabei die serverseitige Verschlüsselung und nutzt den oben erzeugten Schlüssel `wings-key-s3` per Referenz.

Die automatisierte Sicherheitsregel würde den IST-Zustand, hier das Terraform Listing, auf die Existenz der genannten Direktive sowie der Konfigurationsanweisung `apply_server_side_encryption_by_default` in einer Ressource vom Typ `aws_s3_bucket` prüfen. Wenn diese Direktive vorhanden ist, entspricht diese dem SOLL-Zustand und die Prüfung würde erfolgreich, ohne Fehler, durchlaufen.

Die automatisierte Sicherheitsrichtlinien würde demnach drei Schritte durchlaufen:

1. Eine hierarchische *Suchabfrage* in der Terraform-Quelle auf die relevanten Elemente: eine Ressource vom Typ `aws_s3_bucket` und der Direktive `server_side_encryption_configuration` .

2. Eine *unäre Operation* `exists` wird ausgeführt, ob das Kind-Element `apply_server_side_encryption_by_default` vorhanden ist.

3. *Das Ergebnis* der Prüfung ist ein *boolescher Wert*: WAHR oder FALSCH. Bei WAHR ist der Positivfall eingetreten. Bei FALSCH wird eine Reaktion erzeugt und beispielsweise eine Fehlermeldung ausgegeben oder die weitere Verarbeitung gestoppt.

Man kann sich weitere Verfeinerungen nach gleichem Muster vorstellen. So könnte der verwendete kryptographische Algorithmus (`sse_algorithm`) gegen eigene Sicherheitsvorgaben geprüft werden. Dies ginge aber bereits über die hier genannte Best-Practice `SEC08-BP02` hinaus. Die Konfiguration der *default* Verschlüsselung für EBS und EFS erfolgt analog mit ähnlichen Direktiven.

Bewertung der Quelle Der Grad der direkten Automatisierbarkeit ist noch relativ gering, da keine konkreten Konfigurations- und Prüfschritte vorgegeben werden. Auf Basis einiger Empfehlungen könnten automatisierbare Regeln mit zusätzlichem Aufwand und Spezialwissen aufgebaut werden, wenn durch das API die benötigten Informationen hierfür ermittelt und geprüft werden könnten. Die Empfehlungen aus den Kategorien Netzwerkschutz, Schutz der Rechendienste, Datenschutz und teilweise Detektion bieten Kandidaten für die CSPM-Automatisierung.

5.2.2 AWS Security Reference Architecture

Die AWS Security Reference Architecture (SRA) wurde erstmals 2021 veröffentlicht, zuletzt im November 2022 überarbeitet und umfasst mittlerweile circa 80 Seiten [39]. Die Einleitung und die Grundlagen in den ersten Kapiteln weisen eine starke Überlappung mit den Inhalten aus dem AWS Well-Architected Framework-Security Pillar auf. Das Hauptkapitel beschreibt die Sicherheits-Referenzarchitektur, wie diese von AWS Professional Services vorgeschlagen wird (siehe Bild 15).

Eine der Basisempfehlungen der SRA ist die Gliederung der Architektur in mehrere Organisationseinheiten (OU) mittels mehrerer AWS Accounts innerhalb einer AWS Organisation. Die vorgeschlagene Gliederung ist im Bild 15 ersichtlich. Die dargestellte Sicht fokussiert die Produktionsumgebung. Zusätzlich wären Strukturen für Entwicklung- und Testumgebungen zu etablieren.

Die SRA gibt somit eine prüfbare Strukturvorgabe, welche AWS-Dienste in welcher gemeinsam gruppiert werden sollten. Detaillierte Empfehlungen, wie diese Gruppierungen konkret miteinander interagieren dürfen oder sollen und welche Regeln empfohlen sind, werden nicht gegeben.

Das SRA-Dokument beschreibt in den weiteren Kapiteln auf fast 50 Seiten jeden einzelnen AWS-Dienst und dessen fachliche Funktion. Verweise auf detaillierende AWS-Dokumentationen werden häufig genutzt.

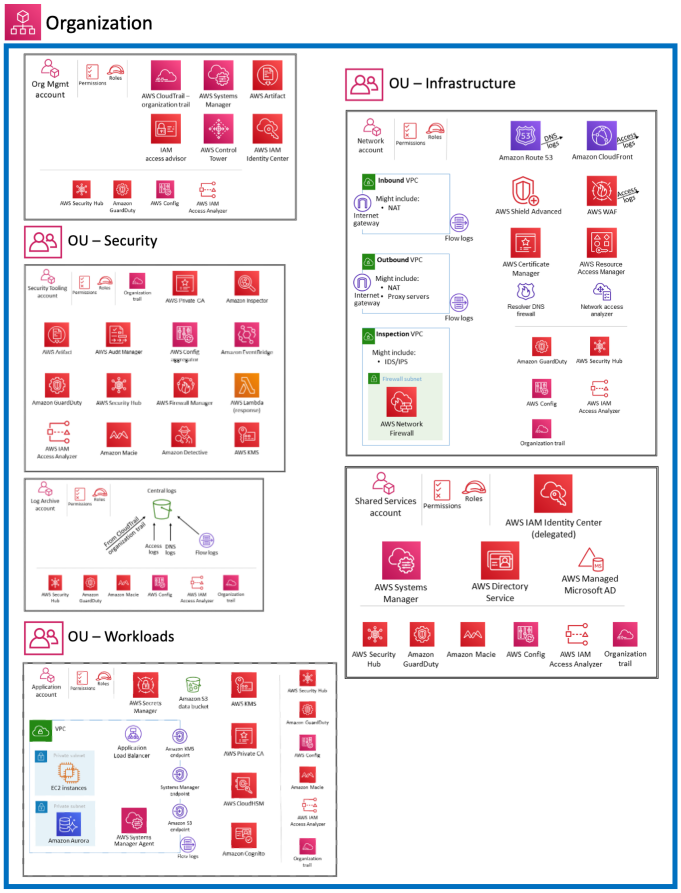

Bild 15: AWS Security Reference Architecture (Quelle AWS [39])

Bewertung der Quelle Aus Sicht der automatisierten Prüfung von Sicherheitsmängeln in der Cloud-Konfiguration sowie einer automatisierten Fehlerkorrektur ergibt sich aus dem derzeitigen Stand des SRA-Dokuments kein Mehrwert für die Entwicklung von Sicherheitsregeln. Die Empfehlungen sind zu abstrakt und zu wenig konkret. Die generelle Gliederung in verschiedene Zonen (OUs) erscheint sinnvoll und könnte prinzipiell mittels CSPM und den API-Methoden zu VPC und Subnetzen geprüft werden.

5.2.3 AWS Security Dokumentation

Die *AWS Security Dokumentation* ist ein Einstieg zu der sehr umfangreichen Sicherheitsdokumention je AWS-Dienst [40]. Die Sicherheitsdokumentation zu den Diensten ist in 28 Kategorien untergliedert. Für die Analyse im Kontext des Cloud Security Posture Management (CSPM) sind speziell die Dienste im Bereich des Infrastructure as a Service (IaaS) relevant. Damit sind aus der *AWS Security Dokumentation*

mehrere Kategorien im Fokus: Compute, Storage, Security Identity & Compliance, Cryptography & PKI, Networking & Content Delivery sowie Management & Governance. Selbst diese eingeschränkte Anzahl an Kategorien umfasst Verweise auf die Sicherheitsdokumentation von 99 AWS Diensten. Aufgrund des sehr hohen Dokumentationsumfang werden hier stellvertretend zwei Dienste als Stichprobe analysiert.

1. *EC2 Security* (Kategorie Compute) auf 117 Seiten mit sechs Schwerpunkten
2. *Amazon S3 Security* (Kategorie Storage) auf circa 320 Seiten

EC2 Security

Die Sicherheitsdokumentation zu EC2 gliedert sich in sechs wesentliche Abschnitte. Der erste Abschnitt behandelt Sicherheitsthemen aus dem Bereich der Netzwerksicherheit, wozu die Themen Segmentierung und Isolation sowie Kontrolle des Netzwerkdatenverkehrs zählen. Das Dokument behandelt die Grundlagen und führt die zugehörigen AWS Dienste und Produkte kurz ein. Konkrete Empfehlungen oder Maßnahmen werden hier nicht behandelt.

Der zweite Abschnitt ist mit *Resilience* betitelt, behandelt jedoch nur knapp die Globale AWS Infrastruktur bestehend aus Regionen, Verfügbarkeits- und lokalen Zonen (siehe Kapitel 4.2.1). Konkrete Empfehlungen oder Maßnahmen werden hier ebenfalls nicht vertiefend behandelt.

Der dritte Abschnitt der EC2 Sicherheitsdokumentation befasst sich mit dem Themenschwerpunkt Datenschutz oder Datensicherheit (englisch data protection). Dieser Abschnitt untergliedert sich in drei weitere Teile. Es wird auf Empfehlungen zu Datenlöschverfahren für den Blockspeicher-Dienst Elastic Block Store aufmerksam gemacht, die dabei dienlich sein können, die Konformität (englisch: compliance) zu Standards wie NIST 800-88 herzustellen. Im zweiten Teil wird die Verschlüsselung von Daten im Ruhezustand (englisch: data at rest) von EBS und auch der Verschlüsselung des Hauptspeichers bei speziellen AWS Diensten behandelt. Es werden weniger konkrete Empfehlungen oder Maßnahmen genannt, sondern es werden Verweise oder Erläuterungen zu Verfahren gegeben. Dies umfasst beispielsweise die aktuell verwendeten Verschlüsselungsalgorithmen wie XTS-AES-256 für NVMe basierte EC2 Instanzen sowie der AWS-Verfahren bei der Schlüsselerzeugung und -speicherung. Der dritte Teil behandelt die Verschlüsselung von Daten

auf dem Transportweg (englisch: data in transit). Auch hier werden Informationen zu verwendeten kryptographischen Verfahren aufgezeigt, Grundprinzipien der Transportverschlüsselung zwischen den AWS Regionen und AWS-Rechenzentren bekannt gegeben sowie verschiedene AWS Produkte im Kontext der gesicherten Datenübertragung genannt. Diese Informationen stellen eine wichtige Grundlage für eine Sicherheitsarchitektur dar, sind aber bisher keine automatisiert prüfbaren Regeln. So ist beispielsweise die generelle Feststellung, dass 'All traffic between AZs is encrypted' (jeglicher Datenverkehr zwischen Verfügbarkeitszonen ist verschlüsselt - auf physischer Ebene) eine wichtige Information, aber stellt keinen prüfbaren Zustand einer Cloud-Konfiguration dar. Prüfbar hingegen wäre, ob der Datenverkehr von oder zu einer EC2 Instanz über einen verschlüsselnden AWS Dienst (z.B. AWS VPN) erfolgt oder nicht. Auch automatisiert prüfbar wäre, ob eine EC2 Instanz die AWS Sicherheitsfunktion der verschlüsselten Kommunikation zwischen EC2 Instanzen nutzt oder nicht. Für die verschlüsselte Kommunikation zwischen EC2 Instanzen stellt die Sicherheitsdokumentation eine konkrete und automatisiert prüfbare Konfiguration bereit, auf Basis der Syntax des AWS Command Line Interfaces (CLI).

Der vierte Abschnitt befasst sich mit der Benutzer- und Zugriffsverwaltung (englisch Identity and Access Management) im Kontext von EC2. Der Abschnitt bleibt zunächst sehr generisch und beschreibt die prinzipiellen Konfigurationsabläufe zur Erstellung von Identitäten und Rollen als auch der Gruppen und Policies. Es werden auch im Weiteren eher Grundlagen vermittelt, als prüfbare Best-Practises aufgezeigt. Die Dokumentation bietet sechs kleinere Beispiele zu IAM Policies, die Berechtigungen zur Verwaltung der EC2 Instanzen darstellen. Es findet teilweise auch eine Wiederholung der Netzwerksicherheit statt.

Der fünfte Abschnitt umspannt des Themenfeld der Verwaltung kryptographischer Schlüsselpaare. Dieser Abschnitt gleicht eher einer Anwenderdokumentation als einem Leitfaden für eine Sicherheitsprüfung. Der Informationsmehrwert für eine CSPM-basierte Sicherheitsautomatisierung ist hier gering.

Der sechste Abschnitt fokussiert die Netzwerk-Firewall Funktion von AWS, die sogenannte Security Group. Dieser Abschnitt hat ebenfalls stärker den Charakter einer Anwenderdokumentation zur Verwaltung von Regelsätzen in der Netzwerk-Firewall. Es werden diverse Arbeitsschritte zur Erzeugung, Aktualisierung und dem Löschen von Regeln aufgezeigt sowie der Umgang mit dem Command Line Interface in diesem Kontext. Im Abschluss der Sicherheitsdokumentation in diesem Abschnitt sind ausführliche tabellarische Übersichten zu typischen Firewallregeln gängiger Anwendungen zu finden (Webserver, Datenbanken, Fernzugriff, AWS EFS, DNS usw.), die

Ports und Protokolle detailliert aufzeigen. Diese können für einen Aufbau automatisierbarer Prüfregeln dienlich sein.

Amazon S3 Security

Die umfangreiche AWS Sicherheitsdokumentation zu Simple Storage Service (S3) kann in die folgenden Sektionen grob gegliedert werden:

1. Datenschutz und Datensicherheit mittels kryptographischer Verfahren
2. Informationssicherheit der Daten auf dem Transportweg von und zu S3
3. Zugriffsschutz mittels IAM für Buckets und Objekte in S3 sowie Eignerschaft der Entitäten.
4. S3 Logging und Monitoring
5. Security Best-Practises für Amazon S3

Die erste Sektion beschreibt Optionen die Daten im Ruhezustand bei AWS S3 zu verschlüsseln. Es wird differenziert zwischen der server-seitigen Verschlüsselung die durch AWS Technologien erfolgt und der Verschlüsselung auf Seiten des Clients. Für letztere Variante wird ein SDK angeboten (Java, .NET, PHP, Go, C++). Für die server-seitige Verschlüsselung können von AWS verwaltete kryptographische Schlüssel genutzt werden oder es können kundenspezifische Schlüssel in AWS erzeugt und genutzt werden. Kunden können auch eigene Schlüssel außerhalb AWS erzeugen und verwalten und diese für die S3-Verschlüsselung einsetzen. Der CSP empfiehlt die Verschlüsselung zu aktivieren und hat seit 5. Januar 2023 auch die Default-Einstellung für neue Buckets und deren Objekte auf server-seitige Verschlüsselung gesetzt.

Die zweite Sektion behandelt die Optionen zur Absicherung der Daten auf dem Transportweg. Die Datenübertragung von und zu S3 Buckets arbeitet über das HTTP Protokoll. Als Transportabsicherung kann daher Transport Layer Security (TLS) gentuzt werden. TLS V1.2 ist möglich, empfohlen wird TLS V1.3. Zusätzlich können weitere Sicherheitsmaßnahmen auf Basis von AWS Diensten ergriffen werden. Es kann eine Site-to-Site VPN Verbindung aufgebaut werden (z.B. zur eigenen Niederlassung) oder es kann auch eine sogenannte AWS Direct Connect Verbindung genutzt werden.

Die dritte Sektion werden die Grundsätze des Zugriffsschutzes auf S3 Buckets und Objekte vermittelt. Wichtig zu verstehen ist, dass grundsätzlich S3 Buckets und Objekte als *privat* gelten und nur der Ersteller (AWS Account) Zugriff hat. Der Ersteller beziehungsweise der Eigner kann dann zusätzlichen Identitäten Zugriff gewähren in dem er Zugriffs-Policies definiert.Eine weitere mögliche aber nicht empfohlene Variante ist die Zurgiffsteuerung über sogenannte Acess control Listen (ACLs). Eine Policy steuert dabei drei Parameter: die Ressource, die Aktion und das Principal (die Rolle, die Identität). Für die Kombination aus den drei genannten Parametern wird ein Effekt definiert. Dieser ist entweder 'zulassen' oder 'ablehnen'. Die Policies werden in einer definierten JSON-Struktur verfasst.

Die vierte Sektion gibt Empfehlungen zur Nutzung spezifischer Protokollierungs- und Überwachungsdienste von AWS (CloudTrail Logs, S3 Access Logs, CloudWatch Alarms) um die S3 Buckets und Objekte überwachen zu können. Dies sind sowohl aus Sicht der opertiven Informationssicherheit als auch der IT-Forensik wichtige Konfigurationseinstellungen um im Falle von Incidents und Angriffsversuchen die notwendigen Informationen zur Analyse und Einleitung zielgerichteter Maßnahmen verfügbar zu haben.

Die fünfte Sektion gibt nun konkrete Handlungsempfehlungen zur Steigerung der Informationssicherheit der AWS S3 Infrastruktur. Dieser Abschnitt gibt knapp 20 Sicherheitsempfehlungen in verschiedenen Themenbereichen, teilweise direkt mit den empfohlenen Parametern zur Anwendung in der Konfiguration oder in S3 Policies.

Bewertung der Quelle Die untersuchten Stichproben der *AWS Security Dokumentation* zeigen prinzipiell eine hohe Relevanz für die Sicherheit der Infrastructure as a Service (IaaS) Cloud-Dienste von AWS und decken diese vollständig ab. Die Charakteristik der Sicherheitsdokumentation gleicht dabei eher einem Grundlagenhandbuch und einer Anwenderdokumentation als einem Sicherheitsleitfaden, der für eine Auditierung und Prüfung der Sicherheitskonfiguration dienlich ist. In eher wenigen Fällen sind markante und konkrete Sicherheitsempfehlungen, die einen prüfbaren Charakter aufweisen, verfasst worden. Ein positives Beispiel ist am Ende der Sicherheitsdokumentation für den S3-Speicherdienst zu finden. In diesem Abschnitt befinden sich 18 direkte Sicherheitsempfehlungen die von dem Einsatz gewissen Überwachungsdienste (AWS Config) bis hin zu prüfbaren Regeln der Datenverschlüsselung reichen. Die Sicherheitsempfehlungen sind noch nicht direkt automatisierbar, wirken aber als Basis für eine Entwicklung automatisierter CSPM-Regelsätze durchaus nützlich. Die Aktualität der Dokumentation kann nur vage bestimmt werden, da

die Änderungshistorie sowohl funktionale wie auch nicht-funktionale Änderungen gemischt protokolliert und die Seitenangaben beim PDF-Export nicht korrekt scheinen.

5.3 Best-Practises behördlicher und gemeinnütziger Organisationen

5.3.1 Bundesamt für Sicherheit in der Informationstechnik

Das IT-Grundschutzkompendium (Edition 2023) umfasst mehr als 800 Seiten und ist in thematische Bausteingruppen untergliedert. Im Kontext der IaaS Cloud-Dienste und automatisierter Prüfverfahren sind nur wenige Bausteingruppen prinzipiell von Relevanz. Hervorzuheben wären die Bausteine aus der Gruppe SYS.1 (IT-Systeme, Server), NET.1 (Netze) und NET.3 (Netzkomponenten), als auch DET (Detektion und Reaktion).

Da das IT-Grundschutzkompendium nicht primär auf virtualisierte IT-Umgebungen ausgelegt ist, sind viele der Anforderungen - aus denen man Maßnahmen zur Absicherung ableiten könnte - in diesem Fall nicht Wert stiftend oder sind eine Aufgabe des Cloud Service Providers.

Es könnten daher nur einzelne Bausteine herangezogen werden, wie beispielsweise NET3.2 Firewall. Beispielsweise könnte die Anforderung aus A2 *Es dürfen keine IT-Systeme von außen über die Firewall auf das interne Netz zugreifen* in maschinenlesbare Regelsätze gewandelt werden und das API genutzt werden, die Konfiguration der virtualisierten Netze zu prüfen. Auch die Anforderung aus A5 *Dabei dürfen immer nur so viele Zugriffsrechte vergeben werden, wie für die jeweiligen Aufgaben erforderlich sind (need-to-know-Prinzip)* kann über eine maschinelle Prüfung der IAM-Berechtigungen zur administrativen Verwaltung der Web Application Firewall Cloud-Ressource ausgewertet werden.

Es ist festzustellen, dass die Aktualität der Quelle als hoch eingestuft werden kann, der Umfang für das CSPM und der Grad der Automatisierbarkeit im Cloud-Kontext in Summe doch als eher mittel bis gering einzuschätzen ist.

5.3.2 Center for Internet Security Inc.

Das CIS ist ein seit circa 20 Jahren bestehender Zusammenschluss von Organisationen und Einzelpersonen, um Materialien und Ressourcen zum Thema Internet-

Sicherheit zur Verfügung zu stellen. Die sogenannten *Benchmarks* finden international eine besondere Anerkennung. Die Benchmarks umfassen Anweisungen und Tipps, um IT-Systeme gegen Bedrohungen abzusichern und werden in einem offenen Community-Verfahren erarbeitet. Neben den Benchmarks, die als Dokumente frei verfügbar sind, bietet CIS gehärtete Betriebssysteme als virtuelle Abbilder an. Diese nennen sich *CIS Hardened Images*. Bei der Erstellung der Abbilder wurden die Empfehlungen der Benchmarks bereits umgesetzt, um ein erhöhtes Sicherheitsniveau zu erreichen.

Sowohl neben den Sicherheitsempfehlungen für verschiedene Betriebsysteme als auch typischer Server-Software oder Datenbanken erstellt die CIS-Community Benchmarks für Cloud-Umgebungen. In diesem Kapitel wird der aktuelle Benchmark mit der Version 1.5.0 für die Cloud-Umgebung von AWS untersucht [42].

Umfang und Grad der Automatisierung Die Empfehlungen des Benchmarks sind in fünf Sektionen gegliedert. In dem Benchmark werden Empfehlungen zu zehn AWS Diensten gegeben, die diesen Sektionen zugeordnet werden können. Je Sektion ist die Anzahl der Empfehlungen in kursiver Schrift dargestellt.

1. Identity and Access Management (IAM) / *21*
    - Identity and Access Management
    - Access Analyzer
2. Storage / *10*
    - Simple Storage Service (S3)
    - Elastic Compute Cloud (EC2)
    - Elastic File System (EFS)
    - Relational Database Service (RDS)
3. Logging / *11*
    - CloudTrail
    - CloudWatch
    - Config
    - S3, KMS, VPC
4. Monitoring / *16*

- CloudTrail
- Security Hub

5. Networking / *5*
    - VPC

Die Sektion *Identity and Access Management* verfügt quantitativ über die meisten Empfehlungen. Diese reichen von sehr einfachen Tipps, wie dem Hinterlegen von Kontaktinformationen über Vorgaben zu Passwort-Regeln und deren Vergabeprozesse, der Berechtigungszuordnung bis hin zu Sicherheitsregeln zur Nutzung von Multifaktor-Authentifizierung (MFA).

Die Sektion *Storage* gibt Empfehlungen je Speicher-Dienst, speziell für die Verschlüsselung der Daten, sowie Möglichkeiten zur Verbesserung des Zugriffsschutzes. Empfehlungen zum AWS Datenbank-Dienst werden auch gegeben, der typischerweise als ein Dienst eines Platform as a Service (PaaS) Angebots eingestuft wird.

In der Sektion *Logging* werden vor allem Empfehlungen für die Aktivierung der Protokollierung mit dem Produkt CloudTrail, sowie zur Protokollierung der Dienste S3 und VPC gegeben. Zusätzlich wird empfohlen in S3 gespeicherte Protokolle mittels kryptographischer Verfahren zu verschlüsseln.

In der Sektion *Monitoring* werden verschiedene Filter, Metriken und Alarmauslöser empfohlen. Zur Verarbeitung werden v.a. die Produkte CloudTrail Logs und CloudWatch Logs eingesetzt. Beispielsweise wird eine Metrik zur Prüfung unauthorisierter API-Aufrufe oder bei Änderung von Routing-Regeln als Empfehlung gegeben.

In der Sektion *Networking* werden Empfehlungen zur Konfiguration der Netzwerk Access Control Listen sowie der SecurityGroups gegeben.

Der Aufbau jeder Empfehlung ist sehr strukturiert. Jede Empfehlung umfasst neben einer kompakten Beschreibung und Begründung der Sicherheitsempfehlung auch eine Verfahrensanweisung zur Prüfung der Cloud-Konfiguration auf den Sicherheitsmangel. Die Verfahrensanweisungen für die Auditierung sind meist in zwei Abschnitte gegliedert. Der erste Abschnitt beschreibt das schrittweise Vorgehen mittels der grafischen AWS-Managementoberfläche. Der zweite Abschnitt dokumentiert die wesentlichen Aufrufe für die AWS-Kommandozeilenumgebung (Command Line Interface (CLI)). Ergänzend können Quellenangaben vorhanden sein sowie Zuordnungen zu den Kontrollgruppen des CIS eigenen Sicherheitsframeworks (CIS Critical Security Controls).

Die Sicherheitsregeln des Benchmarks können als sehr gute Grundlage für die CSPM-Automatisierung eingestuft werden.

Fehlerkorrektur Mögliche Maßnahmen, den Sicherheitsmangel zu beheben, werden ebenfalls strukturiert beschrieben. Meist wird eine schrittweise Verfahrensanweisung gegeben, wie dies über die grafische Web-Managementoberfläche erfolgen kann. Eine direkte Automatisierung, wie dies bei den Prüfregeln über beispielhafte Anwendung der AWS-Kommandozeile gezeigt wird, ist bei der Fehlerkorrektur nicht immer gegeben.

Aktualität Die Benchmarks werden in einem Community-Prozess erstellt. Ergänzungen oder Änderungen werden gemeinsam in einem online-Arbeitsbereich mittels geregeltem Änderungsverfahren vorgenommen. Die Community trifft sich virtuell circa alle zwei Wochen. Releases sind in einem halbjährlichen Takt geplant. Ergeben sich früher wichtige Anpassungen, können Zwischenreleases erzeugt werden. Die Version 1.0.0 wurde vor circa 6 Jahren veröffentlicht und umfasste damals 43 Empfehlungen. Es ist sowohl ein Zuwachs von circa 50% erkennbar als auch die Anpassung und Pflege bestehender Regeln.

5.4 Best-Practises kommerzieller Sicherheitsunternehmen

5.4.1 SANS Institute

Das Escal Institute of Advanced Technologies (SANS) ist vor allem ein Anbieter von Schulungen und Zertifizierungen im Bereich der Informationssicherheit. Das Themenspektrum gliedert SANS in sogenannte Fokusbereiche. Ein Fokusbereich stellt die Cloud-Sicherheit dar. Daneben bilden derzeit u.a. Themen wie Digitale Forensik oder Industrielle Sicherheit weitere Fokusbereiche. Das Curriculum zur Cloud-Sicherheit besteht aktuell aus elf einzelnen Schulungseinheiten. Der Zugriff auf die Schulungsinhalte ist kostenpflichtig und steht dem Autor nicht zur Verfügung.

SANS veröffentlicht zu den jeweiligen Fokusbereichen auch frei zugängliche Dokumente und Sicherheitsempfehlungen. Im Kontext der Sicherheit der Konfigurationen von Public-Cloud stellt der Kurs SEC510 einen Schwerpunkt dar. Zu diesem Kurs hat der SANS-Referent Brandon Evans einen Auszug von Sicherheitsempfehlungen publiziert [43].

Die Veröffentlichung umfasst 26 Seiten, ist eher plakativ aufbereitet und weist den Charakter eines Cheat-Sheets auf. Es werden neun Themenbereiche bearbeitet, wobei zu jedem Bereiche eine Liste mit knappen Stichpunkte gegeben wird. Beispielsweise ist eine Empfehlung *Modify the default VPC Network ACL - Remove the default ingress/egress rules.* Diese Sicherheitsempfehlung ließe sich mit vertretbarem Aufwand in eine API-basierte Prüfregel implementieren. Anderer Sicherheitsempfehlungen sind eher organisatorischer Natur und passen nicht zu dem CSPM-Ansatz.

Bewertung der Quelle Die Veröffentlichung scheint eher den Charakter eines Anreizes zur Buchung eines Trainings zu haben als ein strukturierter und umfängliches Best-Practise Dokument zu Verbesserung der Cloud-Sicherheit zu sein. Der Umfang muss daher als gering eingestuft werden. Der Grad der Automatisierbarkeit der Sicherheitsempfehlungen kann mit mittel eingeordnet werden.

Die Aktualität kann nicht nachvollziehbar geprüft werden, da keine Änderungshistorie erkennbar ist. Der Inhalt wirkt jedoch nicht veraltet und man kann daher von einem noch aktuellen Wissensstand ausgehen.

5.4.2 Aqua Security

Der Softwareanbieter *Aqua Security Software Ltd.* stellt eine öffentlich zugängliche Wissensdatenbank für Schwachstellen nach dem Standard Common Vulnerabilities and Exposures (CVE) und für Fehlkonfigurationen für Cloud-Dienste bereit [44].

Umfang und Grad der Automatisierung Die Wissensdatenbank umfasst für alle gängigen Cloud-Umgebungen (AWS, Azure, GCP, Openstack, Alibaba, Oracle). Die Empfehlungen zur Identifikation von Konfigurationsmängeln ist weiterhin strukturiert nach den Cloud-Diensten der Anbieter. Die Liste der Sicherheitsempfehlung für den AWS-Dienst EC2 umfasst bereits mehr als 250 einzelne Sicherheitsempfehlungen. Die Sicherheitsempfehlungen sind sehr feingranular ausgeprägt, was die große Anzahl begründet. Teilweise werden auch Verweise auf die Quelle der Empfehlung gegeben wie beispielsweise zu Sicherheitsempfehlungen des Cloud-Anbieters AWS. Die Sicherheitsempfehlungen beinhalten oftmals technische Details, wie beispielsweise welche Ports zu prüfen sind und können in der Regel sehr gut als Grundlage für eine automatisierte Sicherheitsprüfung verwendet werden. Für andere AWS-Dienste ist der Umfang an Sicherheitsempfehlungen deutlich geringer. Z.B. werden bei AWS EFS nur drei Empfehlungen gegeben und für den AWS-Dienst S3 29.

Aktualität Die Internetseiten der Aqua Security Datenbank zeigen keine Versionsangaben oder ein Änderungsprotokoll mit Zeitstempel. Eine präzise Beurteilung der Aktualität kann damit nicht erfolgen. Da Aqua Security Inc. selbst ein kommerzieller Anbieter von Cloud Sicherheitslösungen ist kann angenommen werden, dass das Unternehmen wert auf korrekte und aktuelle Informationen legt und für die eigene Unternehmensreputation dies auch in der öffentlichen Datenbank praktiziert.

Bewertung der Quelle Die *aqua vulnerability database*, Sektion *misconfigurations*, wird als sehr umfangreich und detailliert bewertet. Insbesodnere die umfangreichen Beschreibungen zu manuellen Behebung von Konfigurationsmängeln sind bemerkenswert. Teilweise werden die Sicherheitsempfehlungen mit Bezug zu IaC Codesegmenten dokumentiert (CloudFormation und Terraform). Referenzen zu Quellen externer Sicherheitsempfehlungen werden bereitgestellt. Die Quelle wird als sehr geeinigt als Best-Practise Basis für CSPM-Ansatz eingestuft .

5.4.3 Bridgecrew Inc.

Das Sicherheitsunternehmen Bridgecrew Inc. verfügt über eine größere Anzahl von Sicherheitsregeln für Cloud-Dienste und -Ressourcen, die das Unternehmen in einem Online-Katalog als sogenannten *Policy Index* je Cloud-Anbieter veröffentlicht [45]. Die Empfehlungen sind sowohl für IaaS-Ressourcen als auch für PaaS- und ausgewählte SaaS-Dienste wie relationale Datenbanksysteme dokumentiert. Für den CSP AWS sind circa 270 einzelne Regeln aufgeführt. Auch für die Cloud-Plattformen von Google, Microsoft und Alibaba sind ähnliche Sicherheitsregeln verfügbar.

Umfang und Grad der Automatisierung Die Best-Practises aus dem *AWS Policy Index* werden je Sicherheitsregel knapp beschrieben. Bridgecrew verfügt über ein Sicherheitsprodukt auf Basis eines proprietären Python-Frameworks. Durch eine Code-Analyse kann die Logik der Best-Practise manuell erarbeitet werden. Die Regeln aus dem *AWS Policy Index* sind nur sprachlich kurz formuliert, aber ohne technische Details. Aus der Code-Analyse kann ermittelt werden, welche Regeln auf welche Cloud-Ressourcen angewendet werden. Die Regeln sind für das Terraform-Framework und dessen Terraform-Vorlagen Dateien konzipiert. Eine Quellenangabe, woher die Best-Practise abgeleitet wurde oder auf welcher Herstellerempfehlung oder Normenvorgabe diese basiert, ist in der Regel nicht gegeben. Im Quellcode findet

keine weitere Erläuterung statt. Meist umfasst die Regel nur einen Namen mit einer Zeile. Teilweise wird auf die Dokumentation der Cloud-Anbieter verwiesen, um die Umsetzung zu erläutern.

Die Sicherheitsregeln für die AWS-Cloud lassen sich in folgende Kategorien aufteilen. In den eckigen Klammern ist jeweils die Anzahl an verfügbaren Regeln angegeben. Kategorien für Containerdienste und Serverless-Architekturen wurden aus der Liste ausgenommen.

1. Regeln für AWS generell / *112*
2. Regeln für das Identity und Access-Management und andere Policies / *68+5*
3. Regeln für die Protokollierung / *29*
4. Regeln für Netzwerkdienste / *59*
5. Regeln für den Objekt-Speicherdienste S3 / *24*
6. Regeln für das Monitoring / *14*

Der Bereich AWS generell ist quantitativ relativ umfangreich. Dies liegt unter anderem daran, dass hier auch AWS-Dienste behandelt werden, die dem Servicemodell Platform as a Service (PaaS) oder Software as a Service (SaaS) zugeordnet werden können. Beispielsweise für die Verschlüsselung der Daten im Ruhezustand (englisch data at-rest) für die AWS DocumentDB oder die Verschlüsselung der Tabellen in AWS DynamoDB. Im Vergleich zu anderen Quellen, die sich auf Infrastructure as a Service (IaaS) Dienste fokussieren, ist die Menge der Best-Practises damit umfangreicher.

Die Kategorie *Identity und Access-Management und andere Policies* umfasst ähnliche Sicherheitsregeln wie die von CIS veröffentlichten Best-Practises (siehe Kapitel 5.3.2). Die Granularität der Empfehlungen bei Bridgecrew ist weitaus umfangreicher und feiner aufbereitet. Beispielsweise beinhalten beide Kataloge eine Regel zur Vermeidung der Nutzung des Account root Nutzers. Das ist weitgehend identisch in Inhalt und Umfang. Beide Kataloge befassen sich auch mit Vorgaben für Passworte. Der Anbieter CIS gibt beispielsweise in *einer* Regel (Nummer 1.8) vor, dass die empfohlene Länge mindestens 14 Zeichen entsprechen sollte. Der Katalog von Bridgecrew listet 5 einzelne Regeln in diesem Zusammenhang. Das Passwort sollte mindestens 14 Zeichen umfassen, Kleinbuchstaben, Großbuchstaben, ein Symbole, sowie eine Nummer (Regeln BC_AWS_IAM_5, 6, 7, 8 und 9). Diese sehr detaillierte Ausarbeitung in einzelne Regeln kann nützlich sein, eine sehr präzisen Steuerung zu erhalten. Führt

aber auch zu einer künstlich gesteigerten Menge an Regeln und kann einen erhöhten Verwaltungsaufwand bedeuten. Der Bridgecrew Katalog umfasst aber auch neue und wertvolle Regeln, die bisher im CIS-Katalog nicht vorhanden sind. Gerade im Bereich des 'IAM-Housekeeping', der Vermeidung von Berechtigungen die eventuell durch eine unsorgfältiges Berechtigungsmanagement übersehen und nicht entfernt wurden, finden sich interessante Ansätze. Beispielsweise kann die Regel *Ensure unused policies are detached from roles* (BC_AWS_IAM_41) Hinweise auf verwaisten Zuordnungen geben, die ein potentielles Sicherheitsrisiko darstellen können.

In der Kategorie *Netzwerkdienste* sind die Empfehlungen weitaus exakter und inhaltlich auch umfangreicher im direkten Vergleich mit den Regeln von CIS. In weniger Fällen kann hier von einer reinen, erhöhten Feingliedrigkeit ausgegangen werden. Der Katalog der Empfehlungen von Bridgecrew behandelt hier wesentliche mehr AWS Netzwerk-und Sicherheits-Dienste. In Teilen entsteht aber auch hier wiederum eine Menge an Regeln durch die Aufteilung auf einzelne Regeln. Beispielsweise gibt es zehn Einzelregeln, die jede im Grunde nur eine eingehende Firewall-Regel je einem Anwendungsdienst (MongoDB, Kibana, ...) und einem Port definiert. Eventuell ist dies ein Gestaltungsprinzip, das der Autor Bridgecrew vorgibt.

In der Kategorie *Protokollierung* und *Monitoring* sind mehr Ähnlichkeiten festzustellen, wodurch sich der Erkenntnisgewinn weniger stark zeigt im Vergleich zu dem Bereich Netzwerksicherheit.

Der *Grad der Automatisierung* lässt sich nicht direkt aus der Dokumentation erkennen. Dazu ist eine Betrachtung des Quellcodes notwendig. Das Listing 5.2 zeigt exemplarisch einen Ausschnitt aus dem Quellcode einer Prüfregel aus dem Bridgecrew-Framework für die Cloudumgebung von Amazon.

```
class SubnetPublicIP(BaseResourceNegativeValueCheck):
    def __init__(self):
        name = "Ensure VPC subnets do not assign public IP by default"
        id = "CKV_AWS_130"
        supported_resources = ['aws_subnet']
        categories = [CheckCategories.NETWORKING]
        super().__init__(name=name, id=id, categories=categories, supported_resources=
    supported_resources)

    def get_forbidden_values(self):
        return [True]

    def get_inspected_key(self):
        return "map_public_ip_on_launch"

check = SubnetPublicIP()
```

Listing 5.2: Sicherheitsregel aus dem Bridecrew-Framework (SubnetPublicIP.py)

Die Prüfregel `SubnetPublicIP` ist als Python-Klasse definiert und besteht aus drei Methoden. Der Konstruktor `__init__()` setzt die Bridgecrew-Framework spezifischen Metadaten, darunter den Identifizierer `CKV_AWS_130` für die Sicherheitsregel. Die Variable `supported_resources` steuert die durch die Regel zu prüfenden Ressource(n) `aws_subnet` aus der Syntax der Terraform-Vorlage.

Eine Logik zur Ausführung einer Sicherheitsregel ist im Listing 5.2 zunächst nicht direkt erkennbar. Es sind keine typischen Vergleiche mittels wenn-dann Logik ersichtlich.

Die Klasse `SubnetPublicIP` erweitert die Superklasse `BaseResourceNegativeValueCheck` aus dem Bridecrew-Framework, die selbst wiederum die Superklasse `BaseResourceCheck` erweitert. Die Superklasse stellt die einfache Prüflogik bereit. Die Klasse prüft, ob der Schlüssel `inspected_key`, den die Methode `get_inspected_key()` zurück liefert, *nicht* in der Liste der verbotenen Werte `forbidden_values` enthalten ist. Die Methode `get_forbidden_values()` liefert diese Liste an die Superklasse. Die Regel in Listing 5.2 prüft somit, ob die Ressource `aws_subnet` einen Schlüssel `map_public_ip_on_launch` besitzt und dieser nicht `True`, also wahr und gesetzt, ist. Wenn dieser Schlüssel wahr ist, wird allen EC2 Instanzen in diesem Subnetz automatisch eine öffentliche IPv4 Adresse zugewiesen. Dies soll die Regel prüfen und als Fehler berichten.

Das Gegenstück wäre die Superklasse `BaseResourceValueCheck`. Diese prüft mit einer Positivliste und nutzt dazu die Methode `get_expected_values()` (nicht im Beispiel-Listing enthalten).

Der Umfang ist als groß einzustufen. Es ist aber auch anzumerken, dass manche Sicherheitsregeln prinzipiell ähnlich sind, sich nur durch Optionen oder Parameter unterscheiden (beispielsweise bei Ports).

Fehlerkorrektur Eine automatisierte Fehlerkorrektur ist im Quellcode nicht verfügbar. Das Bridecrew-Framework bietet im Python-Konstruktor einen Parameter namens `guideline`, der eine Zeichenkette als manuelle Anweisung aufnehmen könnte. Dieser ist aber in der Regel nicht gefüllt. Das Unternehmen stellt ergänzend zu dem Quellcode eine Dokumentation der Regeln bereit [45]. In diesem Policy-Index wird eine Erläuterung der jeweiligen Sicherheitsregel gegeben, sowie auch Anweisungen zur Fehlerkorrektur. In der Dokumentation gibt es teilweise eine knappe Aussage,

was zu vermeiden ist, oder es liegt eine Verfahrensanweisung für die grafische AWS Management-Console vor. Der Bezug zwischen dem Quellcode und der Dokumentation kann über den Identifizierer `id` aus den Metadaten im Konstruktor hergestellt werden.

Aktualität Analysiert man die Best-Practises der Bridecrew auf Basis des Quellcode-Repositories auf GitHub, zeigt sich eine breite Altersspanne. Es sind Regeln vorhanden, die seit mehr als zwei Jahren nicht verändert wurden sowie vereinzelt neuen Python-Dateien mit einer Aktualisierung vor wenigen Wochen [46]. Die Dokumentation, der Policy-Index, verfügt über keine extern nachvollziehbare Änderungshistorie. Hier steht in der Regel nur „*Updated over 1 year ago*“, selten innerhalb der letzten Monate. Die Sammlung der Best-Practises auf GitHub wirkt durchaus noch aktiv, wird durch eine Gruppe von ca. 300 Personen erweitert und gewartet.

Bewertung der Quelle Die Best-Practises von Bridge Crew Inc. können als sehr relevant und passend für den CSPM-Ansatz gewertet werden. Sowohl der Umfang als auch der Detaillierungsgrad der Informationen ist passend. Jedoch sind die Sicherheitsregeln erst durch eine zeitaufwändige Codeanalyse detailliert erschlossen werden, was einen deutlichen Nachteil für die Übertragbarkeit in andere Technologien bedeutet. Die Aktualität ist Quelle ist ebenfalls überwiegend ausreichend um als Basis für einen Prototypen genutzt werden zu können.

5.5 Bewertung der Best-Practises und Quellen

Die bewährten Methoden der unterschiedlichen Quellen wurden anhand von vier Dimensionen bewertet. Die erste Dimension *Grad der Automatisierbarkeit* bewertet, inwieweit sich die Empfehlungen prinzipiell automatisieren lassen könnten. Die zweite Dimension *Transformationsaufwand zur Automatisierung* gibt eine erste Einschätzung, welcher Aufwand entstehen kann, um Empfehlungen der Quellen zu automatisieren. Die dritte Dimension *Umfang / Vollständigkeit IaaS* beurteilt wie umfänglich die bewährten Methoden die typischen Infrastructure as a Service (IaaS) Funktionen und Dienste abdecken. Die vierte und letzte Dimension *Aktualität* betrachtet, wie oft die Empfehlungen aktualisiert oder ergänzt werden und gibt dazu eine Bewertung.

Je Dimension wird eine Bewertung in definierten Stufen gegeben. Es werden in der Regel drei bis vier Stufen genutzt, die von *gering* bis *hoch* reichen können. Die detaillierten Bewertungskriterien und -stufen sind im Anhang A.1 verfügbar.

Die Ergebnisse der Bewertung ist in Bild 16 dargestellt. Es lässt sich erkennen, dass fast alle Quellen eine durchaus hohe Aktualität aufweisen. Die Nutzbarkeit für eine effiziente Automatisierung liegt vor allem bei den Quellen von *BridgeCrew* und *Aqua Security*. Die Quellen des Cloud-Anbieters AWS sind in dem Dokumentenbereich *AWS Security Documentation* in Teilen auch als zielführend zu bewerten, wobei der Rechercheaufwand zur Identifikation um ein vielfaches höher ist. Die Quelle der Organisation *Center for Internet Security* ist ebenfalls positiv zu bewerten.

Quellen	Bewertung			
	Grad der Automatisierbarkeit	Transformationsaufwand zur Automatisierung	Umfang / Vollständigkeit IaaS	Aktualität
Cloud Provider				
AWS Security Reference Architectur	NICHT AUSREICHEND	--	GERING	AKTUELL
AWS Well-Architected FW	MITTEL	MITTEL	GERING/MITTEL	AKTUELL
AWS Security Documentation	MITTEL	MITTEL	HOCH	UNDEFINIERT
Azure Quellen				
Kommerzielle Sicherheitsunternehmen				
SANS Inc.	MITTEL	MITTEL	GERING	AKTUELL
Bridge crew oder Aqua Security	HOCH	GERING	HOCH	AKTUELL
Behörden und gemeinnützige Organisationen				
BSI IT-Grundschutzkatalog + TR	GERING	MITTEL	MITTEL	AKTUELL
Center for Internet Security	HOCH	GERING	MITTEL	AKTUELL (!)
NIST 800-53	GERING	MITTEL	MITTEL	AKTUELL

Bild 16: Bewertung der Best-Practises Quellen für die CSPM-Automatisierung

6 Prototypische Umsetzung

In diesem Kapitel wird eine prototypische und lauffähige Umsetzung einer Cloud Security Posture Management (CSPM)-Automatisierung erarbeitet. Die erarbeitet Lösung soll Sicherheitsrisiken automatisiert in einer Cloudumgebung eines fiktiven Unternehmens erkennen können. Dazu werden ausgewählte Sicherheitspolicies auf Basis der Best-Practises aus Kapitel 5 erstellt und automatisiert. Das fiktive Unternehmen wird im folgenden Abschnitt beschrieben sowie die Szenarien in denen die Sicherheitspolicies automatisiert angewendet werden.

Die Umsetzung der prototypischen Softwarelösung kann auf unterschiedlichen technischen Wegen erfolgen. Eine erarbeitete Auswahl an Lösungswegen wird aufgezeigt und eine Bewertung der Alternativen wird durchgeführt. Eine der Varianten wird auf Basis der Cloudumgebung des Anbieters AWS umgesetzt.

6.1 Beschreibung fachliches Szenario

Das fiktive Unternehmen ist ein in Deutschland ansässiges produzierendes Unternehmen mit einer eigenen Abteilung zur Produktentwicklung (RnD). Das Unternehmen ist ausschließlich für Kunden im europäischen Raum tätig, somit sind die gängigen Normen hinsichtlich des Datenschutzes (EU-DSGVO) einzuhalten. Das Unternehmen plant eine AWS basierte Cloud-Infrastruktur aufzubauen. Die neue Infrastruktur soll sowohl öffentlich zugänglichen Anwendungen und Daten als auch zur Speicherung sensibler und unternehmenskritischer Daten dienen. Die Personalabteilung plant ebenfalls die AWS Cloud zur zentralen Datenablage der Personalakten zu nutzen.

Die vom Unternehmenssitz arbeitende IT-Abteilung hat eine Architektur konzipiert und diese mittels IaC auf Basis von Terraform modelliert. Das Bild 17 veranschaulicht eine schematische Übersicht der gestalteten AWS Infrastruktur. Die schematische Darstellung zeigt sowohl die Netzwerkstruktur, gegliedert in VPCs und Subnetze, Netzwerksicherheitskompontenten wie Security Groups (SGs) (rot umrandet), die virtualisierten EC2 Instanzen als auch die Datenspeicher in Form von Elastic

File System (EFS) Netzwerkdateisystemen und Simple Storage Service (S3) Objektspeichern. Die IT-Abteilung sieht auch einen BastionHost (oder JumpHost) als zentralen Einstiegspunkt für die Fernwartung der Cloud-Systeme vor.

Die Abteilung für Informationssicherheit hat auf Basis von Best-Practises, wie in Kapitel 5.5 erarbeitet, einen Satz erster Sicherheitsregeln definiert (siehe Tabelle 4). Die neue AWS Infrastruktur soll gegen diese Sicherheitsregeln automatisiert geprüft werden, da zu erwarten ist, dass in Zukunft weitere Rechendienste und Datenspeicher in die AWS Cloud migriert werden.

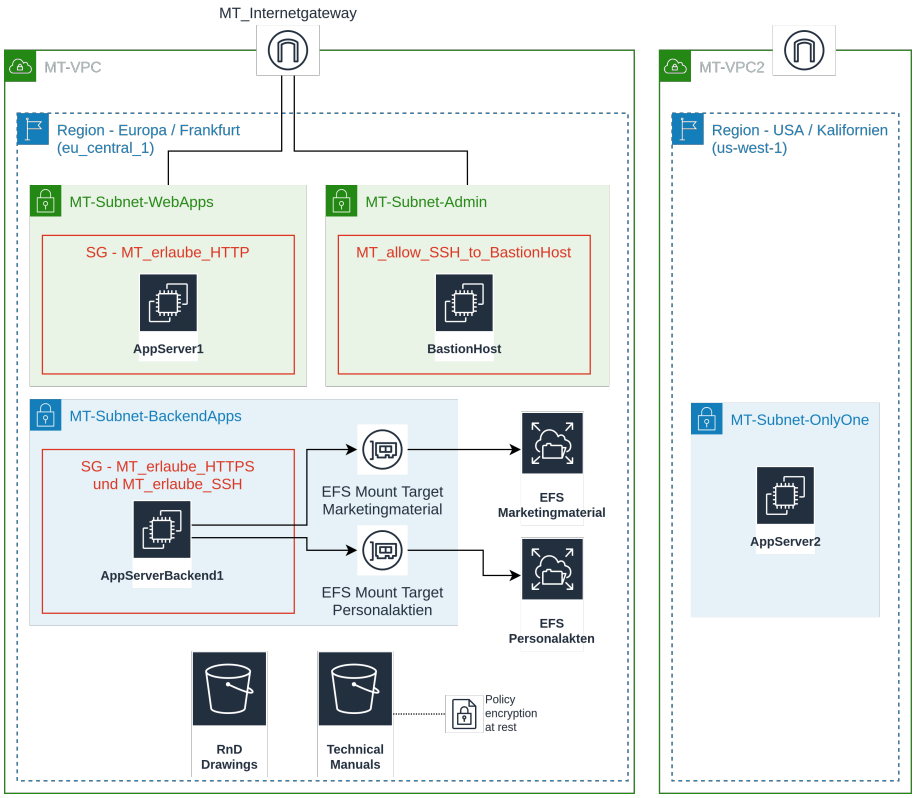

Bild 17: Schaubild AWS Infrastruktur für die prototypische Umsetzung

In das skizzierte Szenario wurden bewusst Sicherheitsmängel eingebaut, die später durch die Sicherheitsregeln automatisiert zu identifizieren sind. So wurden beispielsweise die virtuellen Netzwerke nicht ausreichend segmentiert, Firewalleinstellungen unzulässig offen gestaltet oder Datenspeicher nicht mittel kryptographischer Verfahren abgesichert.

Das fachliche Szenario wurde als Terraform basiertes Infrastructure as Code (IaC) Modell aufgebaut. Der zugehörige Quellcode ist in Kapitel A.5.2 in Listing A.4 dokumentiert.

6.2 Anzuwendende Sicherheitsregeln (Policies)

Die Abteilung für Informationssicherheit hat 15 Sicherheitsregeln als ersten Regelsatz definiert (siehe Tabelle 4). Dabei war es der Abteilung wichtig, sowohl unterschiedliche Best-Practise Quellen zu nutzen als auch wesentliche AWS Ressourcen wie EC2 Rechendienste, Netzwerkspeicher und Blockspeicher zu prüfen. Auch IAM basierte Berechtigungen auf die Dienste sollten geprüft werden. Die Absicherung mittels geeigneter Firewalleinstellungen der Security Groups (SGs) sind ebenso zu prüfen, wie der sichere administrative Zugriff auf die AWS-Ressourcen. Besonders schützenswerten Daten wie beispielsweise die CAD-Zeichnungen aus der Produktentwicklung und die Personalakten sind durch den Einsatz kryptographischer Verfahren abzusichern und geeignete Sicherheitsregeln sind zu wählen.

6.3 Umsetzungsalternativen

Eine prototypische Umsetzung einer Softwarelösung zur automatisierten Prüfung von Sicherheitsregeln - nach dem Prinzip des Cloud Security Posture Management (CSPM) - sollte die wesentlichen Anforderungen aus Kapitel 3.3 prinzipiell abdecken können. Funktionale Defizite sollten durch zusätzliche Softwarebausteine oder Zusatzentwicklungen ergänzt und integriert werden können.

Im Folgenden werden drei unterschiedliche Ansätze zur prototypische Umsetzung skizziert und bewertet. Die Umsetzung erfolgt auf einem der drei Ansätze.

6.3.1 Alternative Entwicklung einer Policy-Analyse Software

Ein möglicher Ansatz wäre die Eigenentwicklung einer Software zur automatisierten Analyse der Cloud-Infrastruktur auf Basis von programmierten Sicherheitsrichtlinien unter Nutzung einer gängigen Programmiersprache. Die Sicherheitsrichtlinien wären dabei in Form eines Quellcodes zu programmieren und würden die zu prüfende Cloud-Infrastruktur auf Basis der Terraform Infrastructure as Code (IaC) Vorlagen analysieren. Die Programmiersprache und deren SDK sollte dazu das Terraform Format entweder nativ oder das Ausgabeformat in Form von JSON verarbeiten können. Das native Terraform Dateiformat und dessen Syntax, die sogenannte HashiCorp Configuration Language (HCL), wird derzeit seitens des Herstellers nur durch ein rudimentäres SDK für die Programmiersprache Go unterstützt.

Nr	Beschreibung	Quelle
1	Use subnets to isolate the tiers of your application (for example, web, application, and database) within a single VPC. Use private subnets for your instances if they should not be accessed directly from the internet.	AWS EC2 Security - Infrastructure [40]
2	Restrict access to your instances using security groups.	
3	Use a bastion host or NAT gateway for internet access from an instance in a private subnet.	
4	Use multi-factor authentication (MFA) with each account.	AWS EC2 Security - Data protection [40]
5	Use SSL/TLS to communicate with AWS resources. We recommend TLS 1.2 or later.	
6	Use AWS encryption solutions [...] Encryption at rest for your EBS volumes and snapshots	
7	Use AWS encryption solutions [...] Remote access encryption, SSH provides a secure communications channel for remote access to your Linux instances	
8	EFS - Enable At Rest Encryption (AVD-AWS-0037, high)	Aqua Security Vulnerability DB - AWS Misconfiguration [44]
9	EC2 - No Secrets In User Data (AVD-AWS-0029, critical). Userdata must not contain access key credentials. Instead use an IAM Instance Profile assigned to the instance to grant access to other AWS Services.	
10	S3 Bucket Encryption Enforcement	
11	S3 Secure Transport Enabled	
12	Ensure no security groups allow ingress from 0.0.0.0/0 to remote server administration ports (5.2)	Center for Internet Security [42]
13	Ensure the default security group of every VPC restricts all traffic (5.4)	
14	Ensure S3 Bucket Policy is set to deny HTTP requests (2.1.2)	
15	Es ist sicherzustellen, dass die EU-DSGVO eingehalten wird. Innerhalb der AWS Infrastruktur sind alle Daten in Europa zu verarbeiten. Dies gilt sowohl für die Speicherung der Daten, als auch für deren Verarbeitung. Der Datentransport hat stets transportverschlüsselt zu erfolgen.	Unternehmenseigene Sicherheitsregel

Tabelle 4: Angewendete fachliche Policies im Rahmen der prototypischen Umsetzung

Die Logik der Sicherheitsregeln wäre damit nicht in Form einer spezifischen deklarativen Sprache abzubilden, sondern würde die typischen Sprachkonstrukte zum Vergleich von IST- und SOLL-Werten nutzen. Die einzulesenden Terraform-Daten (IST-Werte) würden über einen Parser und dessen API untersucht werden. Diese

Parser basierte Suchlogik nach den sicherheitsrelevanten Attributen und Werten, wie diese in der API Analyse in Kapitel 4 erarbeitet wurden, wäre ebenfalls durch Quellcode in der gewählten Programmiersprache abzubilden.

Die Ausgabe der Ergebnisse könnte im einfachsten Fall auf der Kommandozeile erfolgen oder es wäre für eine komfortablere Ergebnisdarstellung ein einfaches Berichtswesen zu programmieren.

Vorteilhaft in diesem Ansatz wäre, dass keine weitere Programmiersprache und Terminologie für ein Unternehmen zu erlernen wäre und gegebenenfalls vorhandene Entwicklerteams die notwendigen CSPM-Policy Logik entwickeln und warten könnten. Bei einem entsprechend größeren Unternehmen und Kapazitätsbedarf könnte diese Entwicklertätigkeit sogar in Länder mit geringeren Lohnkosten ausgelagert werden und so ein ökonomischer Vorteil entstehen. Im Falle der Nutzung des nativen Terraform Formats wäre die Entwicklung jedoch auf Basis der Sprache Go einzuschränken.

Als nachteilig könnte sich erweisen, dass die Erstellung und Pflege stets durch Software-Entwickler/innen zu erfolgen hat und wahrscheinlich die Sicherheitslogik - da diese nicht deklarativ ausgeprägt ist - nur durch die Entwickler:innen verstanden und interpretiert werden kann. Anpassungen und Erweiterungen würden damit immer durch mehrere Personen erfolgen - mindestens den Verantwortlichen für die Sicherheitslogik und den Entwickler:in. Eventuell wäre zusätzlich ein/e Cloud-Infrastruktur-Experte/in notwendig.

Als weiterer Nachteil kann angeführt werden, dass der Ansatz und die entwickelte Softwarelogik nur durch das eine Unternehmen genutzt wird und es unwahrscheinlich ist, dass durch einen Austausch mit anderen Unternehmen ein wirtschaftlicher Vorteil entstehen würde. Der Aufwand und die Kosten für die Entwicklung und Wartung wären wahrscheinlich relativ hoch und diese stets durch das Unternehmen selbst zu tragen.

6.3.2 Alternative dedizierte CSPM Policy-Sprache und Parser

Ein zweiter Ansatz könnte die Konzeption und Entwicklung einer eigenen „Sprache“, also Grammatik und Syntax, für die Gestaltung der Sicherheitssichtlinien und deren Prüflogik sein. Die zu gestaltende Sprache müsste einerseits Fähigkeiten aufweisen die Terraform IaC-Strukturen (Blöcke und Attribute) mittels hierarchischer Suchregeln zu durchsuchen und andererseits müsste die Syntax Sprachelemente aufweisen,

um einen SOLL-IST-Vergleich durchzuführen. Die Rückgabe der Policy-Logik wäre im einfachsten Fall ein boolescher Wert (wahr oder falsch), um die Konformität zur Sicherheitsrichtlinie als Ergebnis zu liefern.

Auch in diesem Fall wäre zusätzlich ein Programm zur Ergebnisaufbereitung als Berichtswesen zu entwickeln.

Als klarer Vorteil ist sicherlich zu nennen, dass eine spezifische Sprache ein deklarativen Ansatz zur Gestaltung der Sicherheitsregellogik darstellt. Die Pflege und Wartung der Sicherheitsrichtlinien wäre, wenn optimal gestaltet, effizient und könnte durch „eine/n“ Experten/in erfolgen. Es könnte eine klare Aufgabentrennung im Unternehmen erfolgen und bräuchte somit keine weiteren Softwareentwickler/innen. Die entwickelte Sprache wäre jedoch sehr spezifisch für den Anwendungsfall CSPM ausgeprägt und müsste damit auch spezifisch erlernt werden. Damit könnten Risiken entstehen, beispielsweise wenn diese Experten/innen nicht ausreichend verfügbar sind, oder ein Unternehmen verlassen.

Nachteilig wäre der große initiale Aufwand für die Konzeption und Entwicklung der Sprache und eines spezifischen Parsers für diese Sprache. Typischerweise würden zunächst Standards wie die Backus-Nauer-Form zur Definition der Sprache zum Einsatz kommen. Zur Verbesserung der Investitionskosten könnten Parser-Frameworks wie beispielsweise LARK genutzt werden [47, 48]. Je nach Ausprägung der Sprache könnte ein erhöhter Wartungsaufwand für den Parser entstehen, wenn zu starke Abhängigkeiten zum API des CSP entstünden und bei Änderung des API Anpassungen am Parser vorgenommen werden müssten.

Auch in diesem Fall würde sehr wahrscheinlich der Nachteil entstehen, dass die erarbeiteten Sicherheitsrichtlinien nicht zwischen Unternehmen ausgetauscht werden könnten, um Kosten und Sicherheitswissen zu teilen.

6.3.3 Alternative Einsatz eines Open-Source Security-Offloader

Das Prinzip der Auslagerung (englisch: offloading) von spezifischen Funktionen auf spezialisierte Dienste, Hard- oder Softwarelösungen, ist aus unterschiedlichen technischen und fachlichen Anwendungsbereichen bekannt und wird praktiziert. Beispielsweise ist das *SSL-Offloading* bekannt, um die TLS-Transportverschlüsselung an einen zentralen, oftmals vorgelagerten Dienst zu zentralisieren, der diese Verschlüsslungsaufgabe zentral für mehrere Applikationsdienste übernimmt. Dies kann

sowohl zu Leistungs- und Kostenvorteilen führen als auch den Betrieb und die Wartung vereinfachen beispielsweise wegen der vereinfachten Schlüsselverteilung.

Auch im Bereich des Zugriffs- und Berechtigungsmanagements können solche Offloading-Szenarien vorteilhaft sein. Anstatt unter Umständen komplexe Regelwerke und Logiken der Zugriffsberechtigung von Benutzern/innen oder zugreifenden Diensten in die jeweiligen Anwendungen direkt zu implementieren, könnte diese Berechtigungsprüfung an einer zentralen Stelle ausgeführt werden. Dies kann unterschiedlich Vorteile aufweisen wie beispielsweise sowohl die einheitliche Definition der Regelwerke mit einer einheitlichen Methodik und Syntax als auch im Rahmen der Wartung zur Pflege der Regelwerke an einem zentralen logischen Ort. Es ist durchaus auch anzunehmen, dass sich die Aktualität und Konsistenz der Berechtigungsregeln dadurch positiv beeinflussen lässt, was zu einer verbesserten Informationssicherheit führen sollte.

Ein Zugriffs- und Berechtigungsmanagements verfolgt dabei eine ähnlich Logik wie diese bei einem CSPM-Ansatz benötigt würde. Anfragen der Zulässigkeit einer Aktion müssen oftmals auf Basis von strukturierten und zusammenhängenden Informationen gegen automatisierte Sicherheitsregeln mit Logikabläufen geprüft werden. Es wird ein erwarteter SOLL-Zustand (Sicherheitsregel) gegen einen anfragenden IST-Zustand (Identität, Daten, Kontext) geprüft. Beispielsweise kann eine Zugriffserlaubnis auf eine Anwendungsdienst nicht nur von einer geprüften Identität des Aufrufenden und dessen Rolle abhängen, sondern auch von dessen genutzten Endgerät, der geografischen Lokation des Aufrufenden, der Tageszeit, dem angefragten Daten- und Anwendungskontext oder diversen anderen Attributen, die zu prüfen sind.

An diesem dritten möglichen Ansatz, auf einem Rahmenwerk eines Offloaders aufzubauen, wäre vorteilhaft dass dieses spezialisiert ist Regelwerke einheitlich technisch zu formulieren und effizient automatisiert auszuführen. Das Rahmenwerk dabei also eine möglichst deklarative Sprache vorgibt und nutzt, die eine Abdeckung der - relativ einfachen - CSPM-Logikbedarfe zur Prüfung der Ressourcenattribute und deren Werte aufweist. Dies würde, wie im Fall des zweiten Ansatzes, die Vorteile der Aufgabentrennung und Effizienz bei der Erstellung und Pflege der automatisierten Sicherheitsregeln erwarten lassen. Ein klarer Vorteil Vorteilhaft gegenüber dem zweiten Ansatz ist, dass eine gewisse Normierung der Sprache und Syntax gegeben wäre. Dies kann dazu führen, dass sowohl die erarbeiteten Sicherheitsregeln zwischen Unternehmen geteilt und wiederverwendet werden könnten als auch die

Investition in den Aufbau der Offloader-Infrastruktur und der Expertise in anderen Sicherheitsbereichen (neben CSPM) eingesetzt werden könnte.

Die Investition in die Konzeption und Entwicklung der Regel-Sprache und eines Parsers würde entfallen, stattdessen könnten bei kommerziellen Produkten Lizenz- oder Nutzungsgebühren anfallen. Je populärer und verbreiteter der Einsatz eines solchen bestehenden Offloaders ist, desto größer ist auch die Wahrscheinlichkeit, Produkt-Experten/innen auf dem Markt beziehen zu können, was die Einarbeitungskosten reduziert. In kleineren Unternehmen oder bei weniger regelmäßigem Anpassungsbedarfen der CSPM-Sicherheitsregeln kann sich auch eine zwischen Unternehmen geteilte externe Teilzeitkraft als Möglichkeit ergeben und für Kosteneffizienz sorgen.

6.3.4 Bewertung der Alternativen

Die möglichen Vor- und Nachteile der drei angedachten Alternativen, wie diese bereits in den vorangehenden Kapiteln ausgeführt wurden, sind in der Übersicht in Bild 18 zusammenfassend dargestellt. Da die drei Alternativen nicht umgesetzt wurden, kann dies als eine Ersteinschätzung gewertet werden.

	Bewertung			
	Abdeckung CSPM Anforderungen	Investitionsaufwand	Abstraktion / deklarativ	Wiederverwendbarkeit
Alternativen				
Eigenentwicklung Policy Software	HOCH	MITTEL / HOCH	GERING	GERING
Dedizierte Policy-Sprache und Parser	MITTEL	SEHR HOCH	SEHR HOCH	GERING
Einsatz Security-Offloader / OSS	MITTEL	GERING / MITTEL	*ZU PRÜFEN*	HOCH

Bild 18: Bewertung der Umsetzungsalternativen

Alle Alternativen erreichen sehr wahrscheinlich eine ausreichende funktionale Abdeckung für einen CSPM-Prototypen. Meist ist zusätzlich in die Ausgabe und ein Berichtswesen zu investieren. Die zweite Alternative sticht mit einer zu erwartenden sehr hohen Investition hervor, ist zusätzlich anzunehmend nachteilig im Bereich der Wiederverwendung im Sinne einer Gemeinschaft über Unternehmensgrenzen hinweg.

Der Einsatz eines „Security-Offloaders“, idealerweise auf Basis von Open-Source, zeigt sich als attraktivster Ansatz und wird im Rahmen dieser Thesis weiter verfolgt.

6.4 Prototyp auf Basis Open-Policy-Agent (OPA)

Im Bereich von Open-Source Softwarelösungen für dieses Anwendungsgebiet gibt es unterschiedliche kleinere und größere Projekte. Ein mittlerweile durchaus bekanntes und verbreitetes Projekt, das seit 2021 als graduiertes Projekt der Cloud Native Computing Foundation (CNCF) geführt wird, ist der sogenannte Open-Policy-Agent (OPA) [49]. Die 2016 entwickelte generische Policy-Engine wurde ursprünglich für das Anwendungsgebiet der Autorisierung und Regelentscheidungen bei Anwendungen (z.B. microservices oder API-Gateways) aufgebaut, das die Berechtigungsprüfung aus der Anwendung extrahiert und zentralisiert. Im Rahmen der Masterthesis wurde die Version 0.49.2 (24. Februar 2023) als Basis genutzt, wobei eine Migration des erstellten Quellcode auf die aktuelle Version 0.52.0 ohne größeren Aufwand möglich sein sollte.

OPA nutzt eine proprietäre, deklarative Regelsprache namens *Rego* um die fachliche Logik der Sicherheitsregeln abzubilden.

Das OPA-Projektteam schreibt dazu selbst:

> *„OPA provides a high-level declarative language that lets you specify policy as code and simple APIs to offload policy decision-making from your software.“*

Die Laufzeitumgebung von OPA besteht aus einem ausführbaren Programm, der OPA-Engine, das sowohl auf der Kommandozeile ausgeführt werden kann als auch in einem Server-Modus mit API/HTTP Schnittstelle genutzt werden kann. Die Sicherheitsregeln (Policies) werden in Form von Dateien erstellt und verwaltet und der Policy-Engine in einem Archivformat - sogenannten *Bundles* - übergeben. Eingehende Informationen mit zu prüfenden Daten werden im JSON Format an die OPA-Engine übergeben. Die Ergebnisse der Regelprüfungen werden ebenfalls im JSON Format zurückgegeben.

In Bild 19 ist dieser Ablauf im Kontext des CSPM-Ansatz schematisch dargestellt. Die zu entwickelnden CSPM-Sicherheitsrichtlinien werden in der Rego-Sprache deklariert und der OPA Policy-Engine übergeben. Einem zu erstellenden CSPM-Client wird die zu prüfende Cloud-Infrastruktur in Form von Terraform basierten IaC Daten übergeben, der diese an die Policy-Engine übermittelt. Die Policy–Engine prüft nun diese eingehenden IaC basierte Cloud-Infrastruktur Vorlage (IST) automatisiert gegen die Sicherheitsrichtlinien (SOLL).

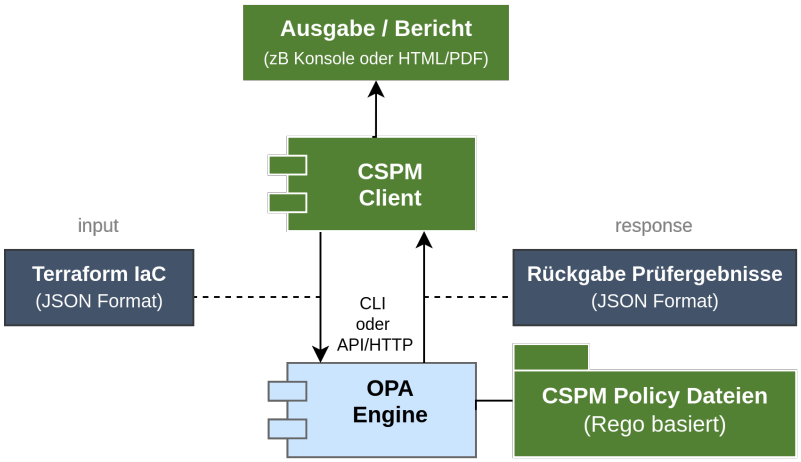

Bild 19: Schematischer Aufbau Open-Policy-Agent im Kontext CSPM

Die Ergebnisse der Sicherheitsprüfung gegen die CSPM-Policies gibt die OPA-Engine in Form von JSON-Daten zurück. Diese JSON-Daten werden aufbereitet, sodass diese für den menschlichen Anwender lesbarer gestaltet sind. Die Ergebnisdaten der Prüfung könnten auch für eine automatisierte Weiterverarbeitung an ein Folgeprogramm weitergeben werden. Beispielsweise an eine aufrufende Continuous Integration/Continuous Delivery (CI/CD) Entwicklungs-Pipline, die in Abhängigkeit der Ergebnisse Aktionen auslöst. Das entwickelte Programm zum Aufruf und der Berichtserstellung ist im Anhang in Kapitel A.6.2 (Listing A.7) dokumentiert.

OPA macht keine festen Vorgaben über den strukturellen Aufbau von Policies. Die OPA Policies sind eher als Funktionen mit oder ohne Rückgabewerte zu verstehen. Im einfachsten Fall gibt die Policy nichts zurück (bleibt leer / ist undefiniert), wenn die Regel nicht positiv durchlaufen wurde. Im positiven Fall ist der 'Rückgabewert' ist ein boolesches TRUE. Im Listing ¸6.1 ist ein solch einfaches Beispiel aufgezeigt. Die Policy `einfachePolicy` durchsucht die eingehenden Daten `input` auf ein Attribut `name` mit dem Wert `Andreas` als Zeichenkette. Ist die Prüfung erfolgreich erhält `einfachePolicy` den Wert `true`. Wenn nicht bleibt `einfachePolicy` undefiniert. Zum besseren Verständnis sind die für einen positiven Fall eingehenden JSON Daten direkt auch im Policy-Listing 6.1 dargestellt.

```
einfachePolicy {
   input.name == "Andreas"
}

# --------------------------------
# Eingehende JSON Daten
{
   "name": "Andreas"
```

```
}
# ---------------------------
```

Listing 6.1: OPA: einfache Rego-Policy

Aufbau der automatisierten Sicherheitsrichtlinien

Die fachliche Struktur der zu automatisierenden Sicherheitsrichtlinien wurde in Kapitel 3.3.2 entwickelt und ist nun bestmöglich technische umzusetzen. Die Rego basierten Sicherheitsrichtlinien für den CSPM-Prototypen sollen dabei einem gleichartigen Aufbau folgenden, sodass der Lösungsansatz durch weitere Sicherheitsrichtlinien sukzessive erweitert werden kann.

Dazu sind drei wesentliche Strukturvorgaben zu definieren:

1. Einheitliche Definition der Informationen zur Sicherheitsrichtlinie
2. Einheitliches Aufrufformat / -struktur der Sicherheitsrichtlinie
3. Einheitliche Struktur der Rückgabewerte

Punkt 1 hat primär die fachlichen Vorgaben aus Kapitel 3.3.2 zu umfassen. Initial wurde im Rahmen der Prototypenentwicklung dazu eine eigene Datenstruktur in Rego aufgebaut, in Form eines Rego *Objekt*s, das aus dem Code der automatiserten Sicherheitrichtlinien aufgerufen und verarbeitet werden konnte. Mit Erscheinen des OPA-Releases v0.52.0 ist in der Dokumentation eine neue Daten-Struktur namens *Metadata* ersichtlich geworden. Dieses Sprachkonstrukt erlaubt es Annotationen in YAML Syntax zu einer Policy zu formulieren. Die Annotationen können damit direkt im Rego-Code erstellt werden und durch den Code selbst ausgewertet werden. Die Metadaten sehen bereits gewisse Attribute vor, die für den CSPM-Prototypen sinnvoll erscheinen und bieten außerdem die Möglichkeit der Erweiterung über sogenannte *custom* Attribute. Es wurde entschieden, den bereits erstellen Masterthesis-Code auf diesen Metadaten-Ansatz umzubauen.

In Bild 20 ist die erarbeitete Zuordnung der fachlichen Vorgaben (links) auf das erweiterte OPA-Metadaten Sprachkonstrukt (rechts) tabellarisch dargestellt. Der ausführliche Code ist in Listing A.8 im Anhang Kapitel A.6.3 verfügbar.

Punkt 2 wurde gelöst durch eine einheitliche Benennung der Policy als `awspolicy[results]` und der Vorgabe eines Namensraums `wings.mtgollwitzer.aws`. Bei der gleichartigen Benennung der Policies wird eine implizite ODER-Funktion der Rego-Sprache

	Beschreibung	OPA METADATA Attribute
ID	<Cloud Kürzel>-<FortlaufendeNummer>	TITLE
Titel	Sprechender Titel	TITLE
Kritikalität	Einstufung angelehnt an NIST CVSS 3.0	CUSTOM.SEVERITY
Beschreibung	Beschreibung des Sicherheitsrisikos [...]	DESCRIPTION
Sicherheitsempfehlung	Beschreibung welche Cloud-Konfiguration empfehlenswert ist [...]	CUSTOM.ADVICE
Prüflogik	`Policy as Code Logik [...]`	*Rego Policy-Code, keine Metadaten*
Behebung	Entweder eine schrittweise Beschreibung zur Behebung [...]	CUSTOM.REMEDIATION bzw. CUSTOM.REMEDIATIONAUTOMATION
Compliance	Verweise auf relevante Kontrollen [...]	CUSTOM.COMPLIANCE
Referenzen	Verweise / Hyperlinks [...]	RELATED_RESOURCES.REF RELATED_RESOURCES.DESCRIPTION

Bild 20: Zuordnung fachliche Vorgaben CSPM-Sicherheitsrichtlinie auf Rego-Metadaten

ausgenutzt, wodurch alle Policies dieses Namens aufgerufen werden und somit nicht jede Policy einzeln aufgerufen werden muss.

```
package wings.mtgollwitzer.aws

awspolicy[results]{
    # Fachliche Logik der CSPM-Sicherheitsrichtlinie
    referenced_ressources = ... # Suche und Vergleiche IST und SOLL
    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
         [...]
    }
    msg := sprintf("Die Ressourcen %s ... verletzen die Regel '%s'", [
    referenced_ressources, annotation.title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources":
    referenced_ressources}, decision}
}
```

Listing 6.2: Vorlage Struktur CSPM-Policy in OPA/Rego

Das Rückgabeformat zu *Punkt 3* wird durch die erarbeitete Datenstruktur `results` abgebildet (siehe Listing 6.2). Diese spezifische Datenstruktur für den CSPM-Prototyp erlaubt es im Fehlerfall sowohl die Informationen zu verletzten Sicherheitsregel (Metadaten in `decision`) zurückzugeben, als auch welche Terraform Ressource zur Verletzung der Sicherheitsregel geführt hat explizit zu benennen (`referenced_ressources`).

Ausführung und Ergebnisse

Die Analyseergebnisse werden durch das erstelle Python-Programm auf der Konsole ausgegeben. In der Übersicht wird dargestellt, welche Sicherheitsrichtlinien, als Policy-Bundle, sowie welche Terraform Plan-Daten verarbeitet wurden. Eine Übersicht der Fehleranzahl und Kritikalität wird im Kopf der Ausgabe aufgezeigt. In Bild 21 ist eine Ausgabe einer CSPM-Analyse dargestellt.

```
andreas@kvm-server:~/cloud.gollwitzer.ch/Studium/Masterthesis/VisualStudioCode/TMP/opa_mtgollwitzer$ /bin/python3 /home/andreas/cloud.gollwitzer.ch/Studium/Masterthesis/VisualStudioCode/TMP/opa_mtgollwitzer/cspm_re
==================================================================================================================================================================
 ANALYSEBERICHT Cloud Security Posture Management | Version 1.0 | MT Andreas Gollwitzer

  Datum:                21/05/2023, 07:17:55
  Terraform Plan:       terraform_plan.json
  Policy Bundle:        bundle.tar.gz
  Anzahl Fehler:        7
    Severity HIGH:      3
    Severity MEDIUM:    4
    Severity LOW:       0
==================================================================================================================================================================
------------------------------------------------------------------------------------------------------------------------------------------------------------------
  Sicherheitsmangel [1/7]: Die S3 Buckets 'aws_s3_bucket.bucket_technical_manuals' verletzen die Regel 'AWS10 - S3 Secure Transport Enabled'. Die S3 Buckets müssen server-seitige im Ruhezustand verschlüsselt sein.

  Policy:                     AWS10 - S3 Secure Transport Enabled
  Kritikalitaet:              HIGH
  Beschreibung:               All statements in all S3 bucket policies must have a condition that requires encryption at a certain level. S3 buckets support numerous types of encryption, including AES-256, KMS us
  Sicherheitsempfehlung:      siehe Masterthesis, Kapitel 4.2.3 Datenspeicher, Abschnitt Objektspeicher. Sowie Tabelle 14 zur AWS CLI zur Prüfung.
  Behebung:                   Configure a bucket policy to enforce encryption.
  Compliance:                 CIS 3.11
  Referenz:                   S3 Bucket Encryption Enforcement
                              https://avd.aquasec.com/misconfig/aws/s3/s3-bucket-encryption-enforcement/
  Referenz:                   Protecting data using server-side encryption with Amazon S3 managed encryption keys
                              https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingServerSideEncryption.html
  Referenz:                   Terraform API Doc - Provides a S3 bucket server-side encryption configuration resource.
                              https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/s3_bucket_server_side_encryption_configuration
------------------------------------------------------------------------------------------------------------------------------------------------------------------
  Sicherheitsmangel [2/7]: Die VPCs {"MT_VPC_2", "VPC_Check_SG_Default", "VPC_Check_SG_Default2"} weisen keine Segmentierung auf und verletzen somit die Regel 'AWS1 - Use subnets to isolate the tiers of your applic

  Policy:                     AWS1 - Use subnets to isolate the tiers of your application within a single VPC.
  Kritikalitaet:              MEDIUM
  Beschreibung:               Das VPC Netzwerk ist ein einzelne Subnetze zu gliedern. Die EC2 Instanzen sind diesen Subnetzen zuzuordnen. EC2 Instanzen dürfen nicht außerhalb von Subnetzen existieren.
  Sicherheitsempfehlung:      Prüfen Sie alle EC2 Instanzen (aws_instance), ob diese einem Subnetz zugeordnet sind. Stellen Sie sicher, dass nur EC2 Instanzen die öffentlich erreichbar sein müssen in public Subne
  Behebung:                   Die EC2 Instanzen (TF Ressource aws_instance) sind mittels dem Parameter 'subnet_id' einem Subnetz zuzuordnen.
  Compliance:                 CIS 12.2, CIS 12.8
  Referenz:                   AWS Security - Infrastructury Security
                              https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/infrastructure-security.html
------------------------------------------------------------------------------------------------------------------------------------------------------------------
  Sicherheitsmangel [3/7]: Die EC2 Instanz(en) 'MT-AppServerBackend1' verletzen die Regel 'AWS9 - No Secrets In User Data'. Die Metadaten UserData dürfen keine sensitiven Daten enthalten.

  Policy:                     AWS9 - No Secrets In User Data
  Kritikalitaet:              MEDIUM
  Beschreibung:               EC2 instance data is used to pass start up information into the EC2 instance. This userdata must not contain access key credentials. Instead use an IAM Instance Profile assigned to t
  Sicherheitsempfehlung:      siehe auch Masterthesis, Kapitel 4.2.2 Rechendienste und Ressourcen, Abschnitt Instanz-Metadaten.
  Behebung:                   Eine nachträgliche Verschlüsselung eines bestehenden EFS Netzwerkspeichers wird derzeit nicht unterstützt. Es ist ein neues verschlüsseltes EFS anzulegen und die Daten sind zu kopier
  Compliance:                 CIS 3.11
  Referenz:                   AWS EC2 - No Secrets In User Data
                              https://avd.aquasec.com/misconfig/aws/ec2/avd-aws-0029/
  Referenz:                   Data encryption in Amazon EFS
                              https://docs.aws.amazon.com/efs/latest/ug/encryption.html
  Referenz:                   Terraform Resource: aws_efs_file_system
                              https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/efs_file_system
------------------------------------------------------------------------------------------------------------------------------------------------------------------
  Sicherheitsmangel [4/7]: Der/die EFS Netzwerkspeicher 'EFS_Personalakten' verletzen die Regel 'AWS8 - Enable At Rest Encryption for Elastic File System (EFS)'. Die Verschlüsselung data-at-rest ist zu aktivieren.

  Policy:                     AWS8 - Enable At Rest Encryption for Elastic File System (EFS)
  Kritikalitaet:              MEDIUM
  Beschreibung:               If your organization is subject to corporate or regulatory policies that require encryption of data and metadata at rest, we recommend creating a file system that is encrypted at res
  Sicherheitsempfehlung:      Sensitive Daten sind durch den Einsatz kryptografischer Verschlüsselung zu sichern. Es ist mindestens die Encryption-at-Rest Funktion von AWS zu nutzen. Des weiteren können die Datei
  Behebung:                   Eine nachträgliche Verschlüsselung eines bestehenden EFS Netzwerkspeichers wird derzeit nicht unterstützt. Es ist ein neues verschlüsseltes EFS anzulegen und die Daten sind zu kopier
  Compliance:                 CIS 3.11
  Referenz:                   EFS Encryption has not been enabled
                              https://avd.aquasec.com/misconfig/aws/efs/avd-aws-0037/
  Referenz:                   Data encryption in Amazon EFS
                              https://docs.aws.amazon.com/efs/latest/ug/encryption.html
  Referenz:                   Terraform Resource: aws_efs_file_system
                              https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/efs_file_system
------------------------------------------------------------------------------------------------------------------------------------------------------------------
```

Bild 21: CSPM Prototyp - Ergebnisdarstellung auf der Konsole

Die jeweiligen Verletzungen der Sicherheitsrichtlinien werden im Detailbereich einzeln aufgelistet. Die Namen der Cloud-Ressourcen, die zur Verletzung der Sicherheitseichlinie geführt haben, werden in der Erläuterung des Sicherheitsmangels explizit genannt. Hinweise, wie der Mangel behoben werden kann und damit die Informationssicherheit der Cloud-Infrastruktur verbessert wird, werden durch eine Erläuterung (Feld Behebung), sowie durch sinnvolle Referenzen auf Dokumentationen zur sicheren Konfiguration bereitgestellt. Die ausgegebenen Detailinformationen in Bild 21 zeigen die Umsetzung der konzeptionellen Vorgaben aus Kapitel 3.3.2 auf Basis der erweiterten OPA-Metadaten aus Kapitel 6.4 dar.

Die bewusst eingebauten Sicherheitsmängel im fachlichen Szenario, siehe Kapitel

6.1, für den CSPM-Prototypen konnten durch die implementierten Rego-Policies detektiert werden.

6.5 Erkenntnisse aus der Umsetzung

Die *funktionalen Anforderungen* zur Detektion der Sicherheitsmängel konnten mit der Rego-Sprache abgedeckt werden. Die Sprache zur Erstellung von automatisierten Analysen von Sicherheitsrichtlinien mit OPA umfasst dabei die strukturierte Suche und Recherche im JSON-basierte Terraform-Plan nach sicherheitsrelevanten Attributen und Wertebereichen. JSON-Syntax Elemente können dabei hierarchisch untersucht werden, was für die geschachtelte Struktur der Terraform-Plan Daten wesentlich ist.

Die funktionale Anforderung der strukturierten Ausgabe identifizierter Sicherheitsmängel, wie in Kapitel 3.3.4 fachlich konzipiert, konnte durch die OPA-Engine selbst nicht abgedeckt werden. Die Rückgabestrukturen mussten spezifisch konzipiert und technisch umgesetzt werden, sowie ein eigenes Programm zur Ausgabe eines strukturierten Berichts entwickelt werden. Der Aufwand dafür ist mit mittel einzustufen, umfasst bisher jedoch nur eine formatierte und menschlich lesbare Ausgabe auf die Konsole. Die Anforderung des automatisierten Auditierungsablaufs, siehe Kapitel 3.3.3, kann als verfügbar angesehen werden. Sowohl die OPA-Engine, als auch das entwickelte Client-Programm, kann zeitgesteuert (z.B. mittels cron-Job) oder durch Ereignisse ausgelöst aufgerufen werden. Die Integration in bestehende Entwicklungsabläufe ist denkbar.

Nach der Einarbeitung konnten Sicherheitsrichtlinien, je nach Komplexität, mit einem *Zeitbedarf* von etwa zwei bis zu sechs oder acht Stunden entwickelt werden. Die Entwicklung wurde in unterschiedlichen Entwicklungsumgebungen ausgeführt. Geprüft wurde die Entwicklung mit Microsoft Visual Studio Code (v1.78.0) mit der Extension 'Open Policy Agent' von Torin Sandall (v0.12.0), als auch mit der sogenannten 'The Rego Playground' Internetapplikation [50]. Ein wesentlicher Nachteil beider Entwicklungsumgebungen ist der Mangel eine Debugging Möglichkeit. Es können zu prüfende Werte stets nur ausgegeben werden, auf die Konsole mittels `print` oder in Hilfsstrukturen in die output-Daten. Rego-Variablen werden auch nicht initial typisiert, wodurch sehr häufig bei der Prüfung der eingehenden JSON-Daten und -Strukturen eine manuelle Typprüfung mittels Hilfsfunktionen während der Entwicklungstätigkeit notwendig ist. Die Transformation der JSON-Objekte und -Typen in Rego-Typen (Sets, Objekte, Arrays, usw.) verlangt leider oftmals ein 'Versuch und

Irrtum' (trial-and-error) Vorgehen. Unabhängig von der Entwicklungsumgebung ist die Rego-Sprache, nicht zuletzt auch durch die vielen impliziten Funktionen, nicht einfach zu erlernen und birgt Fehlerquellen bei kleinen Unachtsamkeiten.

Die Einarbeitungszeit wurde am Anfang als relativ gering angenommen. Diese Annahme konnte nicht bestätigt werden. Die Einarbeitungszeit bis reale Szenarien verlässlich entwickelt werden können ist für den gegeben Kontext mit hoch bis sehr hoch anzusetzen. Die erhöhte Einarbeitungszeit und spätere Effizienz in der Erstellung von Sicherheitsregeln mittels der Rego Programmiersprache wird sich daher vermutlich erst bei häufiger und wiederkehrender Nutzung einstellen können.

Das Projekt Open-Policy-Agent (OPA) gibt an mit Rego eine *deklarative* Sprache zur Entwicklung von Sicherheitsregeln bereitzustellen. Nach der Einarbeitungszeit und idealerweise einem Aufbau spezifischer Hilfsfunktionen können Sicherheitsregeln mit relativ wenig Code-Zeilen erstellt, geprüft und angepasst werden. Aufwändige Code-Sequenzen mit Schleifen, if-then-else Blöcken, Ausnahmebehandlungen usw. entfallen weitgehend, da diese durch minimale Syntaxelemente im Code abgebildet werden. Dies ist einerseits ein technischer Vorteil für den geübten Rego-Entwickler, macht aber die Lesbarkeit des Codes für einen nicht Rego-Experten um so schwerer. Es ist zu erwarten, dass der Rego-Code alleine nicht oder nur sehr wenig dazu beitragen wird dem Anwendungsentwickler oder den Cloud-Infrastrukturexperten ein verbessertes Wissen zur Informationssicherheit zu vermitteln. Die Rego-Policies sind ohne ausführliche Kommentare und ein umfangreicheres Vorwissen um die Terraform Datenstrukturen nicht intuitiv genug. Dies Erwartung wäre an eine deklarativen Sprache zu richten.

Die *Wiederverwendung* von CSPM-Sicherheitsrichtlinien auf Basis von OPA/Rego bleibt prinzipiell ein positiver Faktor für den gewählten Ansatz. Eine rege genutzte und zentrale Plattform zum Austausch von Rego-basierten Sicherheitsregeln scheint sich aber bisher nicht etabliert zu haben. Ein potentiell weiterer positiver Aspekt der Wiederverwendung lässt sich daraus erkennen, dass die erarbeitete Expertise zur Policy-Automatisierung auf Basis der OPA-Technik und der Rego-Sprache auf andere Anwendungsbereiche gut übertragen lässt. Mit dem anhaltenden Trend zur Virtualisierung und der zunehmenden Software-gestützten Definition und Konfiguration von Ressourcen (in der Cloud) kann dies vielfältige Vorteile bringen. Dies ist nicht nur auf den Infrastrukturbereich begrenzt, sondern lässt sich prinzipiell auch auf die sichere Konfiguration von PaaS und SaaS anwenden. Die wesentliche Anforderung ist dabei, dass sich die zu prüfende Konfiguration in Form von JSON-Strukturen abbilden lässt.

Grenzen die sich aufgrund des gewählten Ansatzes, auf Basis der Terraform Plan-Daten vor einem Deployment der Cloud-Infrastruktur zu analysieren, sind erkannbar, wodurch manche Sicherheitsprüfungen nicht ausgeführt werden können. Das liegt weniger an den Fähigkeiten der OPA-Engine oder der Rego-Sprache, sondern an der Datenquelle Terraform-Plan. So kann beispielsweise ein für die Erstellungen einer EC2-Instanz genutztes Amazon Machine Image (AMI) identifiziert werden, es kann aber nicht hinsichtlich des Eigner beziehungsweise des Erstellers geprüft werden. Diese Informationen zum AMI sind nur online in der Amazon Datenbank per API abrufbar und sind nicht in der Terraform-Plan Datei existent.

Es wäre somit für derartig gelagerte Szenarien eine Kombination aus Terraform-Plan Analyse (OPA/Rego), sowie einer ergänzenden online Analyse unter Nutzung des Cloud-API, wie in Kapitel 4 detaillierend ausgeführt, notwendig. Denkbar wäre die zusätzlich benötigten Daten über die OPA *built-in function* `http.send` aktiv während der laufenden Rego-Analyse, somit bei Bedarf, einzuholen. Ein zweites Beispiel sind auch Daten, die nicht in Terraform verwaltet werden, da diese vorab oder bewusst außerhalb Terraform gepflegt werden. Beispielsweise könnten das Daten zu Benutzer und Berechtigungen (Identity und Access Management (IAM)) sein, die in einem zentralen Cloud-Berechtigungsmanagement führend verwaltet werden.

Grenzen hinsichtlich der Skalierbarkeit für sehr große Cloud-Umgebungen konnten im Rahmen der prototypischen Implementierung nicht vertiefend analysiert werden. Die OPA-Architektur lässt aber durchaus ein solides Skalierungsverhalten vermuten, da die Daten und Regeln in Einheiten gegliedert werden könnten, die parallel und unabhängig voneinander effizient verarbeitet werden könnten. Die Laufzeiten in der Testumgebung waren stets in Sekundenbereich, teilweise sogar darunter.

7 Beitrag zur Security-Compliance

7.1 Methodik

Der Mehrwert eines neuen oder ergänzenden Ansatzes zur Verbesserung der Informationssicherheit einer Organisation kann über verschiedene Wege diskutiert und geprüft werden. Sicherheitsstandards und -normen, die über wohldefinierte Anforderungen und Sicherheitskontrollen die Konformität einer Organisation, deren Prozesse und IT-Systeme überprüft, stellt in der Industrie und bei Behörden ein etabliertes Vorgehen dar. In diesem Abschnitt soll die Forschungsfrage aus Kapitel 1.3 aufgegriffen werden, welchen potentiellen Beitrag das Cloud Security Posture Management (CSPM) für die Konformität zu ausgewählten Sicherheitsnormen liefern könnte.

Dazu wurden vier international anerkannte und marktübliche Sicherheitsrahmenwerke stellvertretend selektiert. Die angewendete Vorgehensweise lässt sich aber auch auf andere Normen übertragen. Die Sicherheitsnormen sind dabei bewusst nicht ausschließlich für Cloud-Infrastrukturen relevant, sondern umfassen Informationssysteme, Infrastruktur und dessen Organisation gesamtheitlich. Die Sicherheitsnormen wurden bereits in Kapitel 2.4 vorgestellt.

Die automatisierte Sicherheitsauditierung CSPM basiert auf dem Einsatz automatisierbarer Sicherheitsrichtlinien, die definierte Sicherheitssachverhalte prüfen. Wie beispielsweise, ob ein Datenspeicher eine Datenverschlüsselung der Daten im Ruhezustand nutzt oder nicht. Eine Sicherheitsnorm kann beispielsweise eine solche Anforderung stellen, dass Daten im Ruhezustand oder auch während der Übertragung mittels kryptographischer Verfahren abzusichern sind.

In diesem Kapitel werden die in Kapitel 5 erhobenen Sicherheitsrichtlinien der Unternehmen Bridgecrew Inc. und des Center for Internet Security (CIS) den folgenden Sicherheitsstandards gegenübergestellt:

- Dem Cybersecurity Framework (CSF) (Version 1.1) und der Spezialpublikation SP 800-53 (Rev5) des NIST.
- Dem BSI IT-Grundschutz-Kompendium (2023)

- Den Critical Security Controls (Version 8) der Center for Internet Security (CIS)

Ziel der Gegenüberstellung ist es, den potentiellen Mehrwert der Cloud Security Posture Management (CSPM)-Methodik erkennbar zu machen sowie die Bereiche der Sicherheitsnormen zu identifizieren, in denen der CSPM-Ansatz wirken kann. Die Gegenüberstellung wird in zwei Stufen erarbeitet:

1. Manuelle Einschätzung und Beurteilung der einzelnen Kontrollen oder Anforderungen der Sicherheitsnormen.
2. Teilautomatisierter Ansatz auf Basis einer technischen Analyse unter Nutzung einer Graphendatenbank und der Verweistabellen zwischen den Sicherheitsstandards.

Der erste Schritt nutzt dabei intensiv die gewonnen Erkenntnisse aus den Cloud-API Analyse von Kapitel 4. Diese Erkenntnisse ermöglichen eine Experten/innen-einschätzung, welche automatisierten Sicherheitsanalysen möglich erscheinen. Dieser Schritt erfolgt noch generell und unabhängig von den einzelnen Best-Practises. Die Arbeitsweise und ein Auszug der Ergebnisse ist in Bild 22 dargestellt. Im linken Bildteil ist die manuelle Analyse des NIST CSF ersichtlich, im rechten Bildteil die Bewertung zum BSI IT-Grundschutz-Kompendium.

Aufgrund der großen Menge an automatisierbaren Sicherheitsrichtlinien sowie der Anzahl an Kontrollen der genannten Sicherheitsnormen schien es zielführend, diese Beziehungen in einem Datenbankmodell einander zuzuordnen.

Zur Analyse des möglichen Beitrags der CSPM-Methodik zur Security-Compliance wurde eine Graphendatenbank aufgebaut und zur Auswertung genutzt. Die einzelnen Kontrollen oder Anforderungen der ausgewählten Sicherheitsstandards wurden als Knoten mit deren wesentlichen Merkmalen in Form von Attributen im Graphen modelliert. Die Policies wurden ebenfalls als Knoten mit beschreibenden Attributen angelegt. Zwischen den Knoten vom Typ *Control* und *Policy* wurden Beziehungen angelegt, wenn die Policy eine Beitrag zur Erfüllung des Controls liefern konnte. Die Beziehung wurde durch eine manuelle Analyse der Policies im Abgleich mit den Controls der Standardwerke ermittelt. Dies ist ein relativ zeitintensiver Arbeitsschritt, der später aber in der Analyse des Wertbeitrags zur Konformität sehr hilfreich ist.

In Bild 23 ist ein Ausschnitt des Datenmodells für das CSF des National Institute of Standards and Technology (NIST) und der Policies des Anbieters Bridecrew Inc. für die AWS-Cloud als Übersicht dargestellt. Die grünen Knotenpunkte stellen die

Analyse: NIST Cybersecurity Framework (CSF), Version 1.1

Quelle: https://www.nist.gov/cyberframework/framework

Function	Category & Subcategory		CSPM Relevanz (IaaS Fokus)
IDENTIFY (ID)			
	ASSET MANAGEMENT		
	ID.AM-1	Physical devices and systems within the organization are inventoried	
	ID.AM-2	Software platforms and applications within the organization are inventoried	1
	ID.AM-3	Organizational communication and data flows are mapped	
	ID.AM-4	External information systems are catalogued	
	ID.AM-5	Resources (e.g., hardware, devices, data, and software) are prioritized based on their classification, criticality, and business value	
	ID.AM-6	Cybersecurity roles and responsibilities for the entire workforce and third-party stakeholders (e.g., suppliers, customers, partners) are established	
	BUSINESS ENVIRONMENT		
	ID.BE-1	The organization's role in the supply chain is identified and communicated	
	ID.BE-2	The organization's place in critical infrastructure and its industry sector is identified and communicated	
	ID.BE-3	Priorities for organizational mission, objectives, and activities are established and communicated	
	ID.BE-4	Dependencies and critical functions for delivery of critical services are established	
	ID.BE-5	Resilience requirements to support delivery of critical services are established	
	GOVERNANCE		
	ID.GV-1	Organizational information security policy is established	
	ID.GV-2	Information security roles & responsibilities are coordinated and aligned with internal roles and external partners	
	ID.GV-3	Legal and regulatory requirements regarding cybersecurity, including privacy and civil liberties obligations, are understood and managed	
	ID.GV-4	Governance and risk management processes address cybersecurity risks	
	RISK ASSESSMENT		
	ID.RA-1	Asset vulnerabilities are identified and documented	1
	ID.RA-2	Threat and vulnerability information is received from information sharing forums and sources	
	ID.RA-3	Threats, both internal and external, are identified and documented	
	ID.RA-4	Potential business impacts and likelihoods are identified	
	ID.RA-5	Threats, vulnerabilities, likelihoods, and impacts are used to determine risk	
	ID.RA-6	Risk responses are identified and prioritized	
	RISK MANAGEMENT STRATEGY		
	ID.RM-1	Risk management processes are established, managed, and agreed to by organizational stakeholders	
	ID.RM-2	Organizational risk tolerance is determined and clearly expressed	
	ID.RM-3	The organization's determination of risk tolerance is informed by its role in critical infrastructure and sector specific risk analysis	
PROTECT (PR)			
	IDENTITY MANAGEMENT AND ACCESS CONTROL		
	PR.AC-1	Identities and credentials are managed for authorized devices and users	
	PR.AC-2	Physical access to assets is managed and protected	
	PR.AC-3	Remote access is managed	1
	PR.AC-4	Access permissions are managed, incorporating the principles of least privilege and separation of duties	1
	PR.AC-5	Network integrity is protected, incorporating network segregation where appropriate	1
	AWARENESS AND TRAINING		
	PR.AT-1	All users are informed and trained	1
	PR.AT-2	Privileged users understand roles & responsibilities	
	PR.AT-3	Third-party stakeholders (e.g., suppliers, customers, partners) understand roles & responsibilities	
	PR.AT-4	Senior executives understand roles & responsibilities	
	PR.AT-5	Physical and information security personnel understand roles & responsibilities	
	DATA SECURITY		
	PR.DS-1	Data-at-rest is protected	1
	PR.DS-2	Data-in-transit is protected	1
	PR.DS-3	Assets are formally managed throughout removal, transfers, and disposition	
	PR.DS-4	Adequate capacity to ensure availability is maintained	
	PR.DS-5	Protections against data leaks are implemented	1
	PR.DS-6	Integrity checking mechanisms are used to verify software, firmware, and information integrity	1
	PR.DS-7	The development and testing environment(s) are separate from the production environment	1
	INFORMATION PROTECTION PROCESSES AND PROCEDURES		
	PR.IP-1	A baseline configuration of information technology/industrial control systems is created and	

Analyse: BSI IT-Grundschutz Kompendium (Edition 2023)

Quelle: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/IT-Grundschutz/IT-Grundschutz-Kompendium/IT-Grundschutz-Bausteine/Bausteine_Download_Edition_node.html

Bausteingruppe	Bausteine		Anzahl Anforderungen (generell)	CSPM Relevanz (IaaS Fokus)
ISMS: Sicherheitsmanagement				
	ISMS.1	Sicherheitsmanagement		
ORP: Organisation und Personal				
	ORP.1	Organisation		
	ORP.2	Personal		
	ORP.3	Sensibilisierung und Schulung zur Informationssicherheit	7	1
	ORP.4	Identitäts- und Berechtigungsmanagement	19	1
	ORP.5	Compliance Management (Anforderungsmanagement)	6	
CON: Konzeption und Vorgehensweise				
	CON.1	Kryptokonzept	7	1
	CON.2	Datenschutz	1	1
	CON.3	Datensicherungskonzept	7	1
	CON.6	Löschen und Vernichten		
	CON.7	Informationssicherheit auf Auslandsreisen		
	CON.8	Software-Entwicklung		
	CON.9	Informationsaustausch	9	1
	CON.10	Entwicklung von Webanwendungen		
	CON.11.1	Geheimschutz VS		
OPS: Betrieb				
	OPS.1.1.1	Allgemeiner IT-Betrieb	26	1
	OPS.1.1.2	Ordnungsgemäße IT-Administration	11	1
	OPS.1.1.3	Patch- und Änderungsmanagement		
	OPS.1.1.4	Schutz vor Schadprogrammen		
	OPS.1.1.5	Protokollierung	11	1
	OPS.1.1.6	Software-Tests und -Freigaben	14	1
	OPS.1.1.7	Systemmanagement	25	1
	OPS.1.2.2	Archivierung		
	OPS.1.2.4	Telearbeit		
	OPS.1.2.5	Fernwartung	16	1
	OPS.1.2.6	NTP -Zeitsynchronisation		
	OPS.2.2	Cloud-Nutzung	18	1
	OPS.2.3	Nutzung von Outsourcing		
DER: Detektion und Reaktion				
	DER.1	Detektion von sicherheitsrelevanten Ereignissen	18	1
	DER.2.1	Behandlung von Sicherheitsvorfällen	22	1
	DER.2.2	Vorsorge für die IT-Forensik		
	DER.2.3	Bereinigung weitreichender Sicherheitsvorfälle		
	DER.3.1	Audits und Revisionen		1
	DER.3.2	Revision auf Basis des Leitfadens IS-Revision		
	DER.4	Notfallmanagement		
APP: Anwendungen				
	APP1.1	Office Produkte	11	
	APP1.2	Webbrowser	10	
	APP1.4	Mobile Anwendungen (Apps)	9	
	APP2.1	Allgemeiner Verzeichnisdienst	18	
	APP2.2	AD Domain Service		1
	APP2.3	OpenLDAP	9	
	APP3.1	Webanwendungen und Webservices	11	1
	APP3.2	Webserver	17	
	APP3.3	Fileserver	11	1
	APP3.4	Samba	13	
	APP3.6	DNS-Server	20	1

Bild 22: Stufe 1 - manuelle Beurteilung CSPM-Mehrwert. Beispiel NIST CSF und BSI IT-Grundschutz-Kompendium

108 Kontrollen des CSF dar, die pinken Knotenpunkte visualisieren die etwa 300 Sicherheitsregeln von Bridgecrew Inc. (siehe hierzu auch Kapitel 5.4.3) [45, 18]. Es ist bereits in dieser abstrakten Übersicht erkennbar, dass die Wirksamkeit der Policies sich auf spezielle Kontrollen beziehungsweise deren Kategorien im NIST Framework bezieht. Kontrollen der Sicherheitsnorm (grün), die eine oder mehrere Beziehungen zu Policies (pink) aufweisen, zeigen sich in der Visualisierung durch ein oder mehrere Kanten, wodurch die Kontrollen oftmals im Zentrum zwischen mehreren Policies erscheinen.

Eine detaillierte Sicht einer ausgewählten Kontrolle ist in Bild 24 dargestellt. Es ist auch erkennbar, dass manche Policies zu mehreren Kontrollen Beziehungen aufweisen, d.h. mehrere Kontrollen durch die eine CSPM-Sicherheitsrichtlinie unterstützt werden.

In Bild 23 lässt sich auch ein äußerer Kreis erkennen. Dort befinden sich vor allem die Kontrollen die keine Relation zu einer Policy aufweisen. Diese Kontrollen werden nicht von den CSPM-Sicherheitsregeln unterstützt. Vereinzelt sind auch Sicherheitsregeln (pinke Punkte) erkennbar, die keine Kante aufweisen. Beispielsweise ließ sich die Sicherheitsregel BC_AWS_S3_16 *Ensure AWS S3 object versioning is enabled* nicht sinnvoll einer Kontrolle zuordnen. Daraus lässt sich nun jedoch nicht

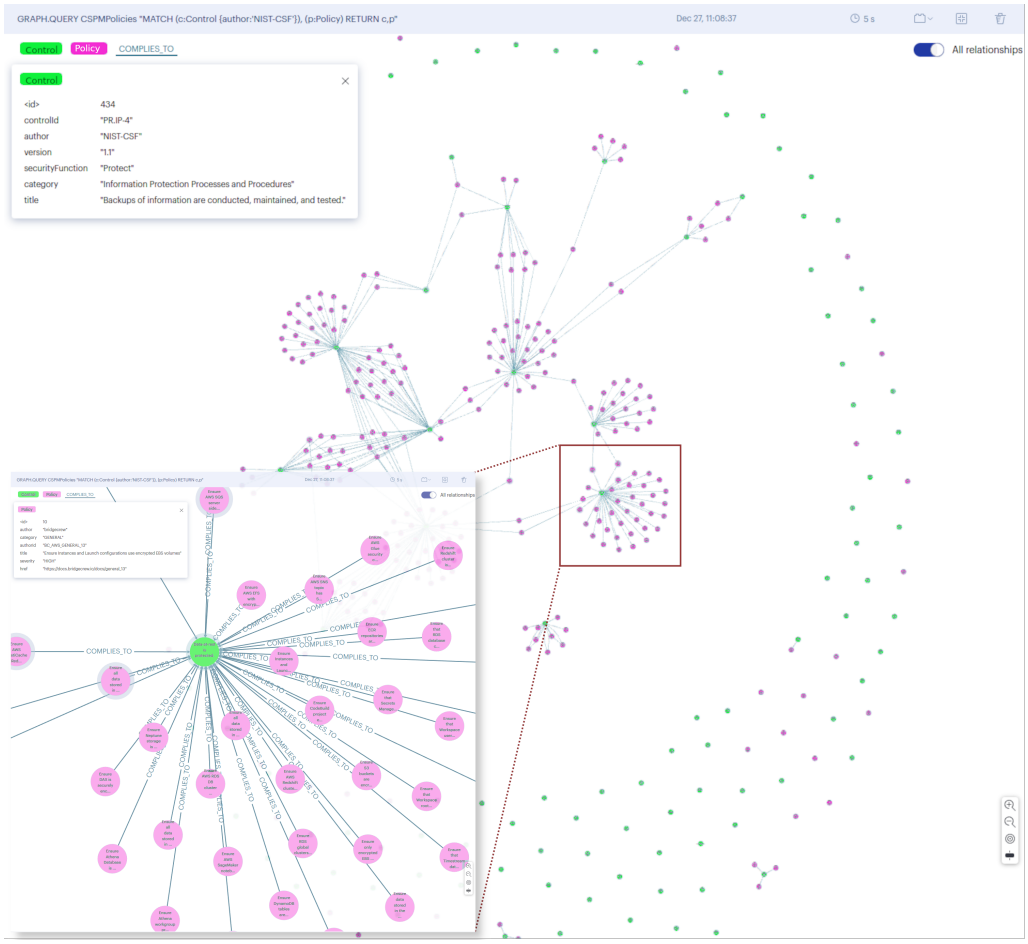

Bild 23: Analyse der CSPM-Policies und Controls über eine Graphendatenbank

schließen, dass diese Sicherheitsregel nicht Wert stiften kann, um die Informationssicherheit in einer Cloud-Umgebung zu verbessern. Es kann eher darauf hindeuten, dass die Sicherheitsnorm in diesem Bereich zu grobe Vorgaben liefert und eventuell eine andere Sicherheitsnorm diesen Aspekt deutlich genauer behandelt und hier eine Zuordnung getroffen werden kann. Diese Sicherheitsregeln sollten in einem realen Informationsverbund einzeln geprüft und nicht vernachlässigt werden.

Zur effizienten Bearbeitung der Relationen zwischen den Policies und den Kontrollen der Sicherheitsnormen im Graphenmodell wurden verfügbare Zuordnungstabellen genutzt [15, 21, 22, 23]. Auf Basis der Zuordnungstabellen können Beziehungen zwischen den Kontrollen aus einem Sicherheitsstandard und den Kontrollen eines zweiten Sicherheitsstandard hergestellt werden. Die Beziehung stellt eine mögliche Gleichartigkeit der Kontrollen dar, wenngleich sich im Detail Unterschiede ergeben können. CSPM-Sicherheitsregeln, die bereits den Kontrollen eines Sicherheitsstandards zugeordnet wurden, können so in einem teilautomatisierten Verfahren wei-

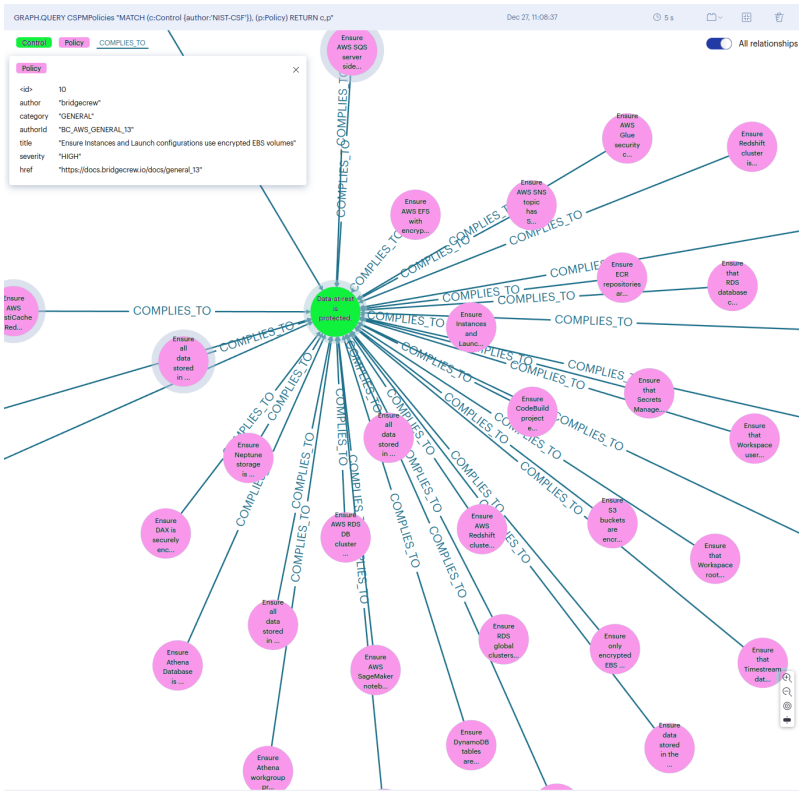

Bild 24: Detailansicht CSPM-Policies von Bridgecrew zu NIST CSF Kontrolle

teren Sicherheitsnormen und deren Kontrollen zugeordnet werden. Eine manuelle Nachprüfung der automatisierten Zuordnung kann die Qualität und Aussagekraft steigern. Gleiches gilt für nicht automatisiert zugeordnete Sicherheitsregeln.

In Bild 25 ist ein Ausschnitt dieses Verfahren mittels Zuordnungstabellen und der Umsetzung in einer Graphendatenbank dargestellt. Die Kontrollen aus den Sicherheitsstandards NIST CSF (rot umrandet) und SP 800-53 (grün umrandet) wurden über die Zuordnungstabelle in Beziehung gesetzt und umgesetzt mit der selbst definierten Kante *MAPS_TO*. Die Beziehung zwischen den CSPM-Policies aus dem Bridgecrew-Katalog und dem Cybersecurity Framework (CSF) wurden bereits im oberen Schritt manuell hergestellt (Bild 24). Durch diese Beziehungskette kann nun eine Relation zwischen den Bridgecrew-Policies und den Kontrollen aus einem neuen Sicherheitsstandard (hier NIST SP 800-53) effizient hergestellt werden.

7.2 Ergebnisse zu den Sicherheitsstandards

Die Ergebnisse der manuelle Beurteilung und aus der teilautomatisierten Analyse wurden zusammengeführt. Die Ergebnisse zu den vier ausgewählten Sicherheitsstan-

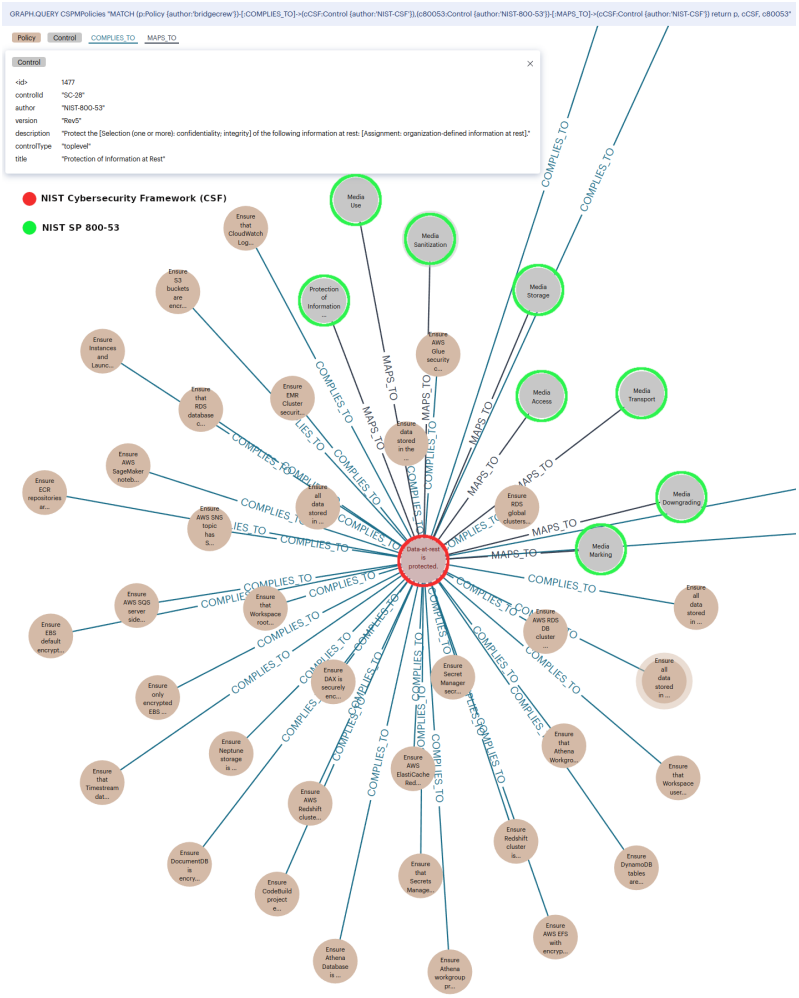

Bild 25: Beziehung zwischen den Controls aus zwei Sicherheitsstandards (NIST CSF und SP 800-53) und den zugeordneten CSPM-Policies (Bridgecrew)

dard sind in den folgenden Kapiteln zusammengefasst.

7.2.1 NIST Special Publication 800-53

Das international anerkannte Standardwerk *Security and Privacy Controls for Information Systems and Organizations* (SP 800-53) zur Steuerung der Informationssicherheit des National Institute of Standards and Technology (NIST) besteht aus über 1.100 Sicherheitsanforderungen bzw. -maßnahmen, die in zehn Kategorien gegliedert sind [16].

Das Ergebnis der Analyse ist in Bild 26 als Übersicht dargestellt.

Die Übersicht in Bild 26 veranschaulicht in welchen der Kategorien die CSPM Me-

SP **800-53** Rev. 5 - Security and Privacy Controls for Information Systems and Organizations

AC | ACCESS CONTROL	PE | PHYSICAL AND ENVIRONMENTAL PROTECTION
AT | AWARENESS AND TRAINING	PL | PLANNING
AU | AUDIT AND ACCOUNTABILITY	PM | PROGRAM MANAGEMENT
CA | ASSESSMENT, AUTHORIZATION, AND MONITORING	PS | PERSONNEL SECURITY
CM | CONFIGURATION MANAGEMENT	PT | PERSONALLY IDENTIFIABLE INFORMATION PROCESSING
CP | CONTINGENCY PLANNING	RA | RISK ASSESSMENT
IA | IDENTIFICATION AND AUTHENTICATION	SA | SYSTEM AND SERVICES ACQUISITION
IR | INCIDENT RESPONSE	SC | SYSTEM AND COMMUNICATIONS PROTECTION
MA | MAINTENANCE	SI | SYSTEM AND INFORMATION INTEGRITY
MP | MEDIA PROTECTION	SR | SUPPLY CHAIN RISK MANAGEMENT

CSPM relevant

Bild 26: Potentieller CSPM Compliance-Beitrag - NIST 800-53

thodik potentiell Mehrwert erzeugen kann. Es ist zu erwarten, dass CSPM den quantitativ größten Mehrwert in den Kategorien *SC - SYSTEM AND COMMUNICATIONS PROTECTION FAMILY* und *RA - RISK ASSESSMENT FAMILY* entfalten kann. In Kategorien die Anforderungen zum Schutz physikalischer oder menschliche Ressourcen stellen (Kategorien PE, PS, MP) oder planerische Aspekte der Informationssicherheit (Kategorien CP, PL, PM, SR) zeigt CSPM keine Vorteile zur Verbesserung der Informationssicherheit. Im Bereich des Identitätenmanagements und der Auditierung (Kategorien AC, AU, CA und CM) zeigt CSPM weiterhin Stärken und kann dazu Beitrage die Informationssicherheit bei der Cloud-Nutzung zu steigern. Die Anforderungen der Norm SP 800-53 in der Kategorie AT sind explizit ausgerichtet auf Fortbildungsmaßnahmen des Personals. In direktem Bezug zu Fortbildung kann CSPM keinen Beitrag liefern.

7.2.2 BSI IT-Grundschutz Kompendium

Das IT-Grundschutz-Kompendium (2023) des BSI verfügt quantitativ über einen gleichartig großen Umfang an Anforderungen an die Informationssicherheit wie NIST SP 800-53. Die in Bild 27 dargestellte Zusammenfassung der potentiellen Mehrwerte durch CSPM für die Konformität zum IT-Grundschutz-Kompendium zeigt auf der obersten Ebene eine relativ starke Abdeckung. In den nicht technischen Bausteingruppen wie beispielsweise *ORP - Organisation und Personal* ist der Beitrag durch CSPM eher indirekt gegeben. Dies kann argumentiert werden indem eine CSPMbasierte Sicherheitsautomatisierung identifizierte Sicherheitsmängel den

Cloud-Nutzern direkt aufgezeigt werden und verbessernde Maßnahmen im Kontext empfohlen werden. Diese Methode kann dazu Beitragen die Sensibilisierung zur Informationssicherheit der Mitarbeiter zu fördern. Auch im konzeptionellen Bereich kann die Entwicklung von Sicherheitsrichtlinien und deren technische Umsetzung zur Automatisierung einen Beitrag zur Informationssicherheit liefern. Der Schwerpunkt des Wertbeitrags liegt aber sicherlich in den Bausteinen im Bereich *NET - Netze und Kommunikation*, als auch *SYS - IT-Systeme/Server* und in Teilen von *APP - Anwendungen*. Einige Bausteine in der Gruppe NET sind auf physische Komponenten und deren Absicherung ausgelegt (z.B. NET3.1 Router und Switches, NET.4.3 Faxgeräte). Diese sind einerseits nicht virtualisiert und liegen andererseits in der Hoheit des CSP und dessen Verantwortung. In diesen Bausteinen kann CSPM keine Vorteile zur Absicherung bieten. Der Bereich Operational Technology (OT) beziehungsweise industrielle IT (IND) ist ebenfalls stark auf physische Anlagen und deren Komponenten ausgerichtet (Aktoren, Sensoren, Maschinen).

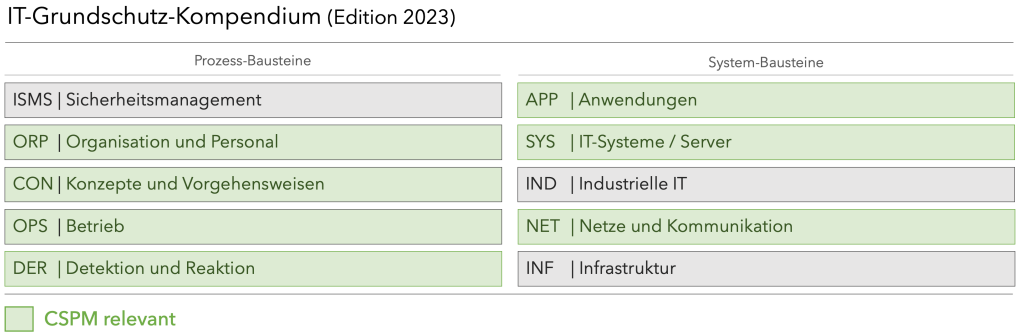

Bild 27: Potentieller CSPM Compliance-Beitrag - BSI IT-Grundschutz Kompendium (2023)

Die konzeptionellen Bausteine (CON) können indirekten Mehrwert durch die CSPM-Methode erfahren. Die Bausteine Kryptokonzept, Datenschutz und Datensicherungskonzept (CON.1 - CON.3) sind primär konzeptionelle ausgerichtet und erarbeiten die Definition beispielsweise geeigneter kryptographische Verfahren. Die spätere Überprüfung der Vorgaben anhand der Cloud-Konfiguration kann mittels CSPM automatisiert geprüft werden.

Es ist anzumerken, dass ein teilautomatisierter Ansatz für das IT-Grundschutz-Kompendium nicht sinnvoll anwendbar war, da beispielsweise keine Zuordnungstabellen zwischen den NIST und dem BSI Sicherheitsstandard verfügbar waren. Ein Zuordnung zwischen NIST CSF/SP 800-53 und ISO 27001/2, sowie ISO 27001/2 und BSI wäre möglich gewesen, würde aber wahrscheinlich zu ungenau werden.

7.2.3 Critical Security Controls (CIS)

Wie bereits in Kapitel 2.4 ausgeführt ist der Sicherheitsstandard von CIS mit 147 Sicherheitsempfehlungen oder -anforderungen sehr kompakt. Dieses fokussierte Format unterstützt die manuelle Prüfung der Sicherheitskonformität zu methodischen Ansätzten. Für die Beurteilung im Kontext CSPM wurde der CIS Controls Cloud Companion Guide (Version 8) genutzt [51]. Da alle Safegurads für IaaS relevant sind war keine spezifische Vorselektion notwendig.

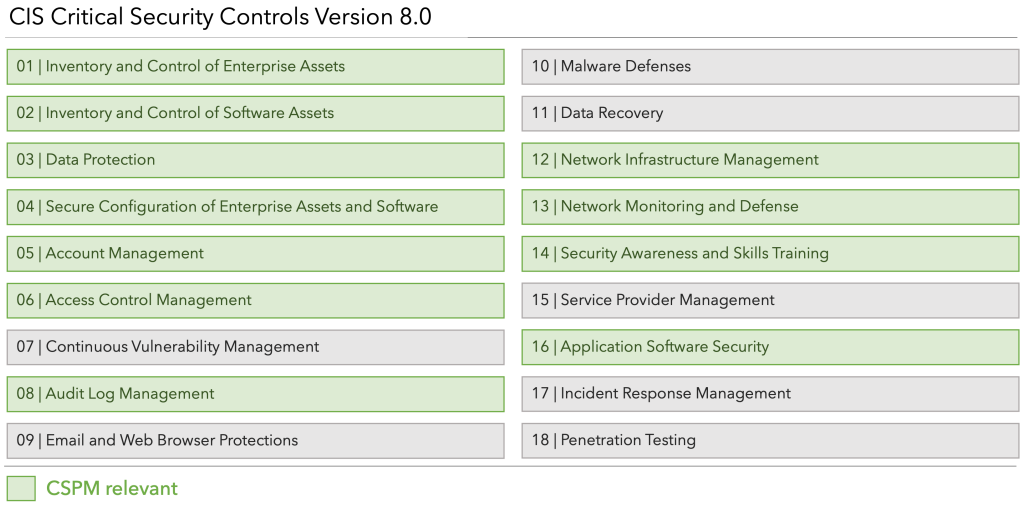

Bild 28: Potentieller CSPM Compliance-Beitrag - CIS Controls Version 8.0 (Mai 2021)

Der Cloud Security Posture Management (CSPM) Ansatz schafft eine relativ breite Abdeckung der 18 Kontrollen des CIS (siehe Bild 28). Schwerpunkt der Unterstützung zur Konformität bietet CSPM in den Netzwerk-Kontrollen 12 und 13 als auch in den grundlegenden Kontrollen zur Inventarisierung (01 und 02). Die Unterstützung im Bereich sicherer Applikationen (16) wirkt zunächst nicht schlüssig, ergibt sich aber aus der Möglichkeit der automatisierten Sicherheitsprüfung von genutzter Infrastruktur (beispielsweise durch gehärtete Betriebssystem-Abbilder). Im Kontrol Audit Log Management kann die automatisierte Sicherheitsprüfung die Aktivierung und Nutzung der Cloud-Logging-Dienste analysieren und prüfen. Im Control 03 - Data Protection sind die Sicherheitsforderungen zur sicheren Verwaltung von Daten im Ruhezustand und der Daten während dem Transport zu finden. Die Abdeckung dieser Anforderungen konnten nachweislich bereits im Rahmen der prototypischen Umsetzung (siehe Kapitel 6) gezeigt werden.

7.2.4 Cybersecurity Framework (CSF) des NIST

Das Cybersecurity Framework des NIST (Version 1.1) ist die Basis vieler anderer Sicherheitsrahmenwerke. Das CSF gibt mit der Struktur aus fünf sogenannter Funktionen, die 97 Sicherheitsanforderungen umfassen, einen Rahmen vor der oftmals durch andere Standards detailliert wird. Dieser Bezug konnte bereits bei dem Standard von CIS erkannt werden.

Cybersecurity Framework (CSF) – Version 1.1, April 2018

ID - IDENTIFY	PR - PROTECT	DE - DETECT
ID.AM | ASSET MANAGEMENT	PR.AC | IDENTITY MGMT AND ACCESS CONTROL	DE.AE | ANOMALIES AND EVENTS
ID.BE | BUSINESS ENVIRONMENT	PR.AT | AWARENESS AND TRAINING	DE.CM | SECURITY CONTINOUS MONITORING
ID.GV | GOVERNANCE	PR.DS | DATA SECURITY	DE.DP | DETECTION PROCESS
ID.RA | RISK ASSESSMENT	PR.IP | INFORMATION PROTECTION P & P	
ID.RM | RISK MGMT STRATEGY	PR.MA | MAINTENANCE	
ID.SC | SUPPLY CHAIN RISK MGMT	PR.PT | PROTECTIVE TECHNOLOGY	

RS - RESPOND	RC - RECOVER
RS.RP | RESPONSE PLANNING	RC.RP | RECOVERY PLANNING
RS.CO | COMMUNICATIONS	RC.IM | IMPROVEMENTS
RS.AN | ANALYSIS	RC.CO | COMMUNICATIONS
RS.MI | MITIGATION	
RS.IM | IMPROVEMENTS	

CSPM relevant

Bild 29: Potentieller CSPM Compliance-Beitrag - NIST CSF (Version 1.1)

Das Analyseergebnis ist in Bild 29 dargestellt. Die Ähnlichkeit der Abdeckung zu dem Analysergebnis bei CIS ist erkennbar. Der CSPM-Ansatz wirkt vor allem in der Funktion Protect (PR) und unterstützt die Funktionen DE - Detect mit kontinuierlicher Sicherheitsanalysen, sowie RS - Respond durch eine automatisierte und zeitnahe Kommunikation identifizierter Schwachstellen in der Cloud-Konfiguration.

7.3 Zusammenfassung

Die Analyse des Wertbeitrags zur Verbesserung der Informationssicherheit durch Cloud Security Posture Management (CSPM) konnte in den ausgewählten Sicherheitsrahmenwerken der NIST, des BSI und der CIS aufgezeigt werden. Die Analyseergebnisse zeigen, dass direkter Mehrwert im Bereich der Ressourcentransparenz, der Netzwerksicherheit und des Datenschutzes zu erwarten ist. Auch im Bereich des Berechtigungsmanagement können Vorteile durch den Einsatz von CSPM erreicht werden und die Konformität zu den Sicherheitsnormen verbessert werden.

Der methodische Ansatz mittels der Kombination aus einer manuellen Experteneinschätzung und dem nachgelagert automatisierten Ansatz unter Nutzung der entwickelten Graphendatenbank konnte ein schlüssiges Gesamtbild erzeugen. Die Methodik bietet die Möglichkeit weitere Sicherheitsstandards, wenn diese eine Zuordnungstabelle zu den bereits genutzten Sicherheitsnormen bieten, mit moderatem Aufwand und sehr effizient zu analysieren. Automatisierte CSPM-Sicherheitsrichtlinien können in diesem Verfahren ebenfalls effizient zu Anforderungen (Kontrollen, Safeguards) aus meherern Sicherheitsnormen zugeordnet werden, wenn diese vorab manuell einem der Standards zugeordnet wurden.

Das automatisierte Verfahren kann als Einstieg der Sicherheitseinordnung und -abschätzung verstanden werden. Das Verfahren wird Schwächen zeigen, wenn der erste zugeordnete Standard einen hohen Abstraktionsgrad aufweist (wie beispielsweise CSF) und die automatisch zugeordneten Sicherheitsnormen wesentlich detaillierter sind (beispielsweise IT-Grundschutz-Kompendium und SP 800-53). Die Anforderungen der Sicherheitsnormen sind im Detail auch nicht identisch formuliert und damit nicht identisch im Sinne der erreichten Konformität zu werten. Eine manuelle Nachprüfung, ob die automatisierte Zuordnung stimmig und korrekt ist, wird stets empfohlen.

8 Zusammenfassung und Ausblick

8.1 Zusammenfassung

Die umfangreiche Analyse der Programmierschnittstelle in Kapitel 4 wurde am Beispiel des Cloud-Anbieters Amazon Web Services (AWS) ausgeführt. Die Ergebnisse der Analyse zeigen deutlich auf, dass das Application Programming Interface (API) umfangreichen lesenden Zugriff auf sicherheitsrelevante Konfigurationsinformationen der Cloud-Infrastruktur einer Kundenumgebung ermöglicht. Die Bestimmung der dazu notwendigen Cloud-Service Operationen, deren Attribute und Werte, verlangt ein tiefgehendes Verständnis der jeweiligen Cloud-Technologie, das vorab zeitaufwendig zu erarbeiten ist. Das API ermöglicht auch den Zugriff auf schreibende und verändernde Operationen, wie beispielsweise das Beenden oder Löschen einer Cloud-Ressource. Diese modifizierenden Operationen bieten die grundsätzliche Möglichkeit bei identifizierten Sicherheitsmängeln in der Cloud-Konfiguration automatisiert einzugreifen.

Die Analyse der Sicherheit der Cloud-Konfiguration mittels des Cloud-API zeigt aber auch Grenzen auf. Das API kann stets nur Konfigurationsinformationen zu bereits installierten und damit in der Regel aktiven Cloud-Ressourcen liefern. Potentielle Sicherheitsmängel können damit nicht vor einer aktiven Nutzung der Cloud-Ressourcen detektiert werden. Es sind auch inhaltliche Grenzen erkennbar. Beispielsweise können Abbilder (englisch: images), die als Vorlage für die Erzeugung eigener virtueller Maschinen genutzt werden, über die AMI-API-Operationen nur sehr oberflächlich analysiert werden. Abbilder die bereits einen erhöhten Sicherheitsschutz bieten oder die durch zertifizierte Quellen mit Sicherheitsnormung herausgegeben wurden, sind nicht nachweislich und unveränderbar auszumachen (siehe Kapitel 4.2.2). Ebenso kann zwar mittels des API ein administrativer SSH-Schlüssel einer EC2-Instanz untersucht werden, dieser kann aber nach der Initialisierung durch den Unix-Administrator ausgetauscht werden, ohne dass das API diese nachträglich Anpassung erkennen kann. Dies kann zu einer trügerischen Sicherheitslage führen die real nicht gegeben ist.

Um eine wesentliche Schwäche der Sicherheitsanalyse mittels des API zu vermeiden wird als Erkenntnis dieser Masterthesis empfohlen Cloud-Infrastrukturen durch Infrastructure as Code (IaC) Techniken zu definieren und zu verwalten (siehe Kapitel 4.3). Diese Code- und Software-gestützte Verwaltung von Cloud-Infrastrukturen bietet die Möglichkeit Sicherheitsmängel bereits vor der Installation und Aktivierung der Cloud-Ressourcen zu detektieren. Dieser Ansatz konnte erfolgreich im Rahmen der prototypischen Umsetzung (siehe Kapitel 6) ausgeführt werden. Der Terraform basierte IaC-Code wurde automatisiert auf Sicherheitsmängel untersucht und es konnten die eingebauten Sicherheitsmängel detektiert werden. Grenzen dieses Ansatzes entstehen vor allem wenn nicht die gesamte Cloud-Infrastruktur durch IaC-basierte Technologien verwaltet wird, wodurch die automatisierte CSPM-Analyse referenzierte Cloud-Ressourcen nicht überprüfen kann. Dies kann beispielsweise bei außerhalb IaC verwalteter Identitäten und Berechtigungen entstehen.

Die im Rahmen der prototypischen Umsetzung genutzte technische Plattform Open-Policy-Agent (OPA) (siehe Kapitel 6.4) konnte die wesentlichen funktionalen Anforderungen (siehe Kapitel 3.3) abdecken. Die entwickelten automatisierten Sicherheitsrichtlinien auf Basis der proprietären Sprache Rego konnten die Sicherheitsmängel in der Cloud-Konfiguration detektieren und dazu passende Maßnahmen zur Verbesserung der Informationssicherheit vorschlagen. Der Einarbeitungs- und Entwicklungsaufwand für die automatisierten CSPM-Sicherheitsrichtlinien ist mit zwei bis zu sechs Stunden je Richtlinie durchaus groß. Und dies zudem bei bereits vorab erfolgter intensiver Analyse des Cloud-API beziehungsweise der Terraform Cloud-Konfiguration. Ein positiver wirtschaftlicher Effekt stellt sich vermutlich erst ein, wenn eine hohe Dynamik der Anpassung und Veränderung der Cloud-Infrastruktur zu erwarten ist oder wenn sich Unternehmen zur Entwicklung dieser OPA-basierten CSPM-Sicherheitsregeln verbinden um die Kosten zu teilen. Derzeit ist (noch) keine Community in diesem Bereich erkennbar.

Die Ergebnisse der Analyse und Bewertung der Best-Practises zur Verbesserung der Informationssicherheit im Bereich Cloud-Infrastruktur zeigen insbesondere bei den kommerziellen Unternehmen Bridge Crew Inc. und Aqua Security Inc. eine bereits sehr gute Grundlage für CSPM. Auch Teile der Sicherheitsempfehlungen des Cloud-Anbieters AWS schaffen ausreichenden Detaillierungsgrad um als Basis zur Entwicklung automatisierter Sicherheitsrichtlinien genutzt werden zu können (siehe Kapitel 5).

Die Methodik Cloud Security Posture Management (CSPM) hat damit das Potential auch einen merklichen Mehrwert zur verbesserten Konformität zu anerkannten

Sicherheitsstandards zu leisten (siehe Kapitel 7.2). Insbesondere in Kategorien des Datenschutzes, der Netzwerksicherheit und des Zugrissschutzes als auch generell im Bereich der Risikoanalyse kann der automatisierte CSPM-Mehrwert schaffen. Indirekt können auch positive Auswirkungen auf die Sensibilisierung und Kompetenzen zur Informationssicherheit der Mitarbeiter erwartet werden, da die Methodik direkte Rückmeldung bei Sicherheitsmängeln an die Entwickler:innen und Administrator:innen liefern kann sowie Maßnahmen zur Verbesserung direkt im Kontext des Mangels bereit stellt.

8.2 Ausblick

Der Cloud Security Posture Management (CSPM)-Ansatz Sicherheitsmängel in der Konfiguration von Cloud-Ressourcen automatisiert zu identifizieren ließe sich prinzipiell auch auf Platform as a Service (PaaS) und Software as a Service (SaaS) Dienste ausdehnen. Eine wesentliche Voraussetzung dafür ist, dass die Cloud-Dienste eine API zur Analyse der Cloud-Dienst Konfiguration anbieten müssen oder die Konfiguration über maschinenlesbare Formate als Code definiert werden. Beispielsweise in Form von JSON-Dateien die maschinell ausgewertet werden könnten.

Eine Fortführung der Masterthesis wäre auch mit Hinblick auf eine nahtlose Integration in automatisierte Softwareerstellungsprozesse denkbar und auch auf die Anwendung von Container-basierten Infrastrukturen.

Eine Wirtschaftlichkeitsbetrachtung der Cloud Security Posture Management (CSPM)-Methodik im konkreten Umfeld ein oder mehrere Unternehmen beziehungsweise Organisationen könnte den Mehrwert des Ansatzes mittels Kennzahlen untermauern.

A Anhang

A.1 Bewertungskriterien Best-Practise Quellen

Die Quellen bewährter Methoden (Best-Practise) wurden auf Basis eines für diese Thesis entwickelten Schemas bewertet. Die Kriterien werden im Folgenden konkretisiert. Das Bild 30 ist eine Kopie aus dem Kapitel 5.5 und wird hier lediglich illustrativ genutzt, damit die Erläuterung der Kriterien besser verstanden werden kann.

Quellen	Bewertung			
	Grad der Automatisierbarkeit	Transformationsaufwand zur Automatisierung	Umfang / Vollständigkeit IaaS	Aktualität
Cloud Provider				
AWS Security Reference Architectur	NICHT AUSREICHEND	--	GERING	AKTUELL
AWS Well-Architected FW	MITTEL	MITTEL	GERING/MITTEL	AKTUELL
AWS Security Documentation	MITTEL	MITTEL	HOCH	UNDEFINIERT
Azure Quellen				
Kommerzielle Sicherheitsunternehmen				
SANS Inc.	MITTEL	MITTEL	GERING	AKTUELL
Bridge crew oder Aqua Security	HOCH	GERING	HOCH	AKTUELL
Behörden und gemeinnützige Organisationen				
BSI IT-Grundschutzkatalog + TR	GERING	MITTEL	MITTEL	AKTUELL
Center for Internet Security	HOCH	GERING	MITTEL	AKTUELL (!)
NIST 800-53	GERING	MITTEL	MITTEL	AKTUELL

Bild 30: Bewertungskriterien Best-Practises Quellen (Kopie)

Grad der Automatisierbarkeit

HOCH

- Regeln beinhalten bereits Pseudo-Code oder API/CLI Anweisungen (zumindest für eine Cloud)
- Regeln beinhalten Detektion und geben mindestens schrittweise Anleitung zur Behebung, oder auch API/CLI Code. Nicht zwingend zu jeder Regel, dort wo sinnvoll.

MITTEL

- Regeln sind mit konkretem Bezug zu den Service(s)-Konfiguration, die zu prüfen sind
- Regeln geben abgrenzbaren SOLL-Zustand vor (Detektion), auch mit Werten
- Regeln müssen keinen Code beinhalten, geben aber sprachlich die Regeln ausreichend konkret vor (ein reines Prinzip ist nicht ausreichend)
- Mitigationsmaßnahmen werden mindestens als Verweis auf weitere Informationsquellen gegeben. Konkrete Verfahrensanweisung oder API ist nicht notwendig

NICHT AUSREICHEND

- keine ausreichende Abgrenzung auf eine konkrete Regel. Sehr allgemeingültig und generell formuliert
- keine ausreichende Abgrenzung auf eine konkrete Regel. Sehr allgemeingültig und generell formuliert
- fehlender Bezug auf Service(s) der IaaS
- Keine messbare/qualifizierbaren Schwellwerte oder Grenzen (SOLL) erkennbar

Transformationsaufwand zur Automatisierung

GERING

- Interpretation und Transformation der Regel auf die eigene Cloud-Umgebung in wenigen Stunden; Einzelperson oder kleines Team.
- Aufwand im Bereich von Stunden für ein lauffähiges 'minimum viable product' (MVP) der automatisierten Regel.

MITTEL

- Konzeptioneller Aufwand und Analyse notwendig (API, weitere Sicherheitsdokumente).
- *Team* von Experten notwendig.
- Aufwand für Konzeption und MVP im Bereich von wenigen Tagen.
- Ein Ergebnis kann sicher erzeugt werden; mindestens für die Detektion.

- Mitigationsmaßnahmen werden mindestens als Verweis auf weitere Informationsquellen gegeben. Konkrete Verfahrensanweisung oder API ist nicht notwendig

HOCH / SEHR HOCH

- Hoher konzeptioneller Aufwand.
- Diverse Analysen, auch technisch, notwendig.
- ggf. Erweiterung des Frameworks notwendig
- Aufwandsschätzung birgt hohe Unsicherheiten, zu viele unklare Einflüsse.

Umfang / Vollständigkeit IaaS

GERING

- Quantitativ eher wenige Regeln.
- Regeln in nur einem oder wenigen IaaS-Funktionsbereichen.
- Vor allem Regeln zur Detektion.

MITTEL

- Regeln verteilen sich auf mehrere IaaS-Funktionsbereiche.
- Je Bereich sind es mehrere Regeln.
- Jedoch keine vollständige Abdeckung aller IaaS-Funktionsbereiche.
- Regelumfang sowohl einfache Regeln (Attribute), als auch erste komplexere Regellogiken (über mehrere Services im Verbund).
- Erste Regelgrundlagen zur Mitigation/Fehlerbehebung vorhanden, oder schrittweise Arbeitsanweisungen.

HOCH

- Vollständige Abdeckung aller IaaS-Funktionsbereiche bzw. zu den wesentlichen IIaaS-Services.
- Im Vergleich zu anderen Best-Practise Quellen ist die Quantität der Regeln im oberen Drittel.
- Regelsätze umfassen mehrere komplexe Regeln.

- Mitigationen, wenn sinnvoll, in der Mehrzahl vorhanden (auch wenn noch nicht automatisiert).

Aktualität

UNDEFINIERT

- Es konnte keine klare Änderungshistorie nachvollzogen werden.

AKTUELL

- Änderungshistorie kann nachvollzogen werden.
- Letzte Aktualisierung liegt weniger als 12 Monate zurück.
- ein regelmäßiger Aktualisierungsprozess scheint vorhanden, ein aktives Team / ein aktives Unternehmen scheint vorhanden zu sein.

NICHT AKTUELL

- Änderungshistorie kann nachvollzogen werden.
- Letzte Änderung liegt deutlich über 12 Monate zurück und/oder
- Ein aktives Team scheint nicht mehr vorhanden zu sein.
- Regelwerk hat den Charakter eines Archivs.

A.2 Zuordnungstabelle AWS-Regionen

In der Tabelle 5 ist eine Übersicht der AWS Regionen (Stand Dezember 2022) dargestellt. Der Code je Region wird dem Staat bzw. Land zugeordnet. Falls seitens AWS der der Bundesstaat und die Stadt dokumentiert ist, wird dies ebenfalls zugeordnet. Als Informationsquelle diente die AWS Dokumentation zu EC2, Abschnitt Regionen und Zonen der Netzwerkfunktionen [52]. Für Informationen zu AWS Diensten in der Volksrepublik China ist ergänzend die spezifische Dokumentation zu nutzen [53].

Code	Staat	Bundestaat	Stadt
us-east-1	USA	Virginia	-
us-east-2	USA	Ohio	-
us-west-1	USA	Kalifornien	-
af-south-1	Südafrika	-	Kapstadt
ap-east-1	VR China	Hong Kong	Hong Kong
cn-north-1	VR China	-	Peking
cn-northwest-1	VR China	Ningxia	-
ap-southeast-3	Indonesien	-	Jakarta
ap-south-1	Indien	-	Mumbai
ap-northeast-3	Japan	-	Osaka
ap-northeast-1	Japan	-	Tokyo
ap-northeast-2	Südkorea	-	Seoul
ap-southeast-2	Australien	-	Singapur
ap-southeast-3	Indonesien	-	Sydney
ca-central-1	Kanada	-	-
eu-central-1	Deutschland	-	Frankfurt
eu-west-1	Irland	-	-
eu-west-2	UK	-	London
eu-south-1	Italien	-	Milan
eu-west-3	Frankreich	-	Paris
eu-north-1	Schweden	-	Stockholm
me-south-1	Bahrain	-	-
me-central-1	VAE	-	-
sa-east-1	Brasilien	-	São Paulo

Tabelle 5: Zuordnung AWS Regionen zu Staaten und Städten

A.3 AWS API Analyseergebnisse

A.3.1 Übersicht der AWS IaaS-Dienste

Die Tabelle 6 zeigt eine Gesamtübersicht der wesentlichen AWS Dienste im Bereich Infrastructure as a Service (IaaS). Dienste die mit Stern gekennzeichnet sind, wurden nicht im Rahmen dieser Thesis behandelt.

IaaS-Funktionsbereich	AWS Dienstnamen (Kürzel)
Basisdienste und Strukturen	AWS Account und Organization
	Region, Verfügbarkeitszone (Availability Zone) und lokale Zone
	Resource Groups *
Rechendienste (Computing)	EC2 Instance, Network-Interface, Volume
	Amazon Machine Image
	Instance-Metadatenservice (IMDS)
	AWS Systems Manager *
	EC2 Image Builder *
	AWS Outpost *
Datenspeicher und -sicherung	Elastic Block Store
	Elastic File System
	Simple Storage Service
	AWS Backup mit Backup Gateway *, Amazon S3 Glacier *
	Storage Gateway * / S3 File Gateway *
Netzwerk und -Sicherheit	EC2 Virtual Private Cloud (VPC), Subnet, Route / RouteTable
	Route 53 (DNS)
	CloudFront (CDN) *
	Shield (DDoS)
	Elastic Load Balancing (ELB): Application-, Network- und Classic-Load Balancer *

IaaS-Funktionsbereich	AWS Dienstnamen (Kürzel)
	Internet Gateway
	Virtual Private Network (VPN)
	Direct Connect *
	Transit Gateway *
	API Gateway *
	NetworkAcl
	SecurityGroup
	Network Firewall
	Web Application Firewall (WAFv2)
	Network Manager *
	SecurityHub *
	Amazon GuardDuty und Detective *
Schlüsselverwaltung	Key Management Service (KMS)
	CloudHSM *
	EC2 KeyPair
Identitäts- und Zugriffsverwaltung	IAM
	Access Control List (ACL)
Monioring und Logging	CloudWatch / CloudWatch Logs *
	CloudTrail *
FaaS und Container-Dienste	EC2 Container Service (ECS) *
	Elastic Kubernetes Service (EKS) *
	EC2 Container Registry (ECR), ECR private registry *
	Lambda *

Tabelle 6: Übersicht potentiell CSPM relevante IaaS- und FaaS-Cloud-Dienste von AWS

A.3.2 API Regionen und Verfügbarkeitszonen

Die Tabelle 7 ist das Ergebnis der API-Analyse aus Kapitel 4.2.1.

API: `EC2.DescribeAvailabilityZones und EC2.DescribeRegions`		
Pfad/Objekt	**Schlüssel**	**Werte und CSPM-Relevanz**
Region[]	regionName	Zeichenkette mit dem Bezeichner (Code) der Region, Beispiel `eu-central-1`. Syntax folgt der Aufstellung aus Tabelle 2. Frage(n): 1-3
AvailabilityZone[]	zoneName	Zeichenkette mit dem Bezeichner der Verfügbarkeitszone (Availability Zone), Beispiel `eu-central-1a`. Syntax folgt der Aufstellung aus Tabelle 2. Frage(n): 1-3
	zoneId	Zeichenkette mit der technischen Kennzeichnung der Verfügbarkeitszone. Beispiel: zoneId `euc1-az2` entspricht der Verfügbarkeitszone mit zoneName `eu-central-1a`. Wie erkennbar, lässt sich die technische Kennzeichnung nicht einfach aus dem Namen der Verfügbarkeitszone ableiten (az*2* versus central-1*a*). Frage(n): 1-3
	zoneType	Es werden drei Typen unterschieden: `availability-zone`, `local-zone` und `wavelength-zone`. `Wavelength` ist speziell für 5G-Netzwerk gestützte Anwendungen, wiederum um niedrige Latenz zu erreichen. Frage(n): 3

Tabelle 7: API Objekte - Geographische Information bzw. Rechenzentren

A.3.3 API EC2 Instanz

Die Tabelle 8 ist das Ergebnis der API-Analyse aus Kapitel 4.2.2.

A.3.4 API Abbilder auf Basis Amazon Machine Images

Die Tabelle 9 ist das Ergebnis der API-Analyse aus Kapitel 4.2.2.

A.3.5 API Instanz Metadaten

Die Tabelle 10 ist das Ergebnis der API-Analyse aus Kapitel 4.2.2.

A.3.6 API Datenspeicher - Blockspeicher

Die Tabelle 11 ist das Ergebnis der API-Analyse aus Kapitel 4.2.3.

A.3.7 API Datenspeicher - Netzwerkdateisystem

Die Tabelle 12 ist das Ergebnis der API-Analyse aus Kapitel 4.2.3.

A.3.8 API Datenspeicher - Objektspeicher

Die Tabelle 13 ist das Ergebnis der API-Analyse aus Kapitel 4.2.3.

A.3.9 API Virtualisiertes Netzwerk und Netzwerksicherheit

Die Tabelle 14 ist das Ergebnis der API-Analyse aus Kapitel 4.2.4.

A.3.10 API Schlüsselverwaltung

Die Tabelle 15 ist das Ergebnis der API-Analyse aus Kapitel 4.2.5.

A.4 Graphendatenbank für Best-Practise Policies

Aus einer Auswahl von Sicherheitsregeln, aus den Quellen der Best-Practises, wurde eine Graphendatenbank für CSPM aufgebaut. Jede Sicherheitsregel stellt dabei einen Knoten in der Graphendatenbank dar und trägt wesentliche Informationen in Form von Eigenschaften (englisch properties). Die Regeln werden den AWS-Ressourcen zugeordnet, die selbst auch als Knoten abgebildet sind. Die Beziehungen werden über Relationen (englisch edges oder relationsship) abgebildet. Gleichartige Sicherheitsregeln aus unterschiedlichen Quellen werden ebenfalls miteinander über Relationen in Verbindung gebracht. Die Datenbank ist erweiterbar. So können weitere Informationen zu den Regeln ergänzt werden, neue Regeln hinterlegt werden, weitere Relationen ergänzt werden usw.

Die Sicherheitsregeln wurden weiterhin in Bezug zu ausgewählten Sicherheitsstandards und Compliance-Regeln gebracht. Die Relationen zwischen den Sicherheitsregeln und den Controls aus den Sicherheitsstandards unterstützt die Konzeption und Analyse, welche Regeln einen Mehrwert zur Verbesserung der Konformität bieten kann.

Die im Rahmen dieser Thesis aufgebaute Graphendatenbank für CSPM ermöglicht eine strukturierte Suche über mehrere Best-Practise Quellen. Sicherheitsregeln können beispielsweise auf Basis der Kritikalität gesucht werden. Sicherheitsregeln können auch in Bezug zu der zu schützenden Cloud-Ressource selektiert werden.

In einer weiteren Ausbaustufe könnten auch die automatisierten Regelsätze, als Vorlagen, hinterlegt werden.

A.4.1 Technisches Datenmodell

```
GRAPH.QUERY CSPMPolicies "
# Policies / Sicherheitsregeln anlegen (AWS Security, Bridgecrew, CIS, Aqua Security,
    Eigene ...)
CREATE
  (:Policy {
    author:'bridgecrew',
    authorId:'BC_AWS_GENERAL_1',
    category:'GENERAL',
    title:'Ensure EC2 instances have tags',
    severity:'LOW',
    href:'https://docs.bridgecrew.io/docs/general_1'}
  ),[...]
  (:Policy {
    author:'cis',
```

```
    authorId:'1.1',
    version:'1.5.0',
    category:'IAM',
    title:'Maintain current contact details',
    description:'Ensure contact email and telephone details for AWS accounts are current
     and map to more than one individual in your organization.',
    rational:'If an AWS account is observed to be behaving in a prohibited or suspicious
     manner, AWS will attempt to contact the account owner by email and phone using the
    contact details listed. If this is unsuccessful and the account behavior needs
    urgent mitigation, proactive measures may be taken, including ...',
    href:'https://workbench.cisecurity.org/sections/844438/recommendations/1387855'}
  ), [...]

# Controls anlegen (NIST 800-53, NIST CSF, CIS 8, ...)
CREATE
  (:Control {
    controlId:'AC-2(1)',
    author:'NIST-800-53',
    version:'Rev5',
    description:'Support the management of system accounts using ...',
    controlType:'sublevel',
    title:'Account Management',
    titleDetail:'Automated System Account Management'})
  },
  (:Control {
    controlId:'ID.AM-1',
    author:'NIST-CSF',
    version:'1.1',
    securityFunction:'Identify',
    category:'Asset Management',
    title:'Physical devices and systems within the organization are inventoried.'}
  ),[...]
```

Listing A.1: Cypher Query Language - Erzeugen der CSPM Datenbank

```
GRAPH.QUERY CSPMPolicies "
MATCH (c:Control {controlId:'ID.AM-2'}),
      (p:Policy {authorId:'BC_AWS_GENERAL_1'})
      CREATE (p)-[:COMPLIES_TO]->(c)"

GRAPH.QUERY CSPMPolicies "
MATCH (c80053:Control {controlId:'PM-5', author:'NIST-800-53'}),
      (cCSF:Control {controlId:'ID.AM-1', author:'NIST-CSF'})
      CREATE (c80053)-[:MAPS_TO]->(cCSF)"
```

Listing A.2: CSPM Datenbank - Relationen zwischen Policies und den Controls, sowie zwischen den Controls anlegen

A.4.2 Technische Umsetzung

Als technische Plattform wurde die in-memory Datenbank *Redis* mit dem Zusatzmodul *RedisGraph* in der Version 6.2.6 genutzt. Zur grafischen Visualisierung wurde

RedisInsight genutzt, das im Paket *Redis Stack* enthalten ist.

A.5 Terraform Infrastructure-as-Code

A.5.1 Terraform Zustandsmodell

Terraform nutzt zur Verwaltung der Cloud-Infrastrukturen ein Zustandsmodell. Das in Bild 31 dargestellte Zustandsmodell stellt die fünf Zustände dar. Ergänzend sind in Bild 31 die wesentlichen Infrastructure as Code (IaC) Dateien dargestellt, sowie die logische Beziehung zur Cloud-Umgebung des Cloud Service Provider (CSP), sowie zu der Sicherheitsauditierung auf Basis Cloud Security Posture Management (CSPM).

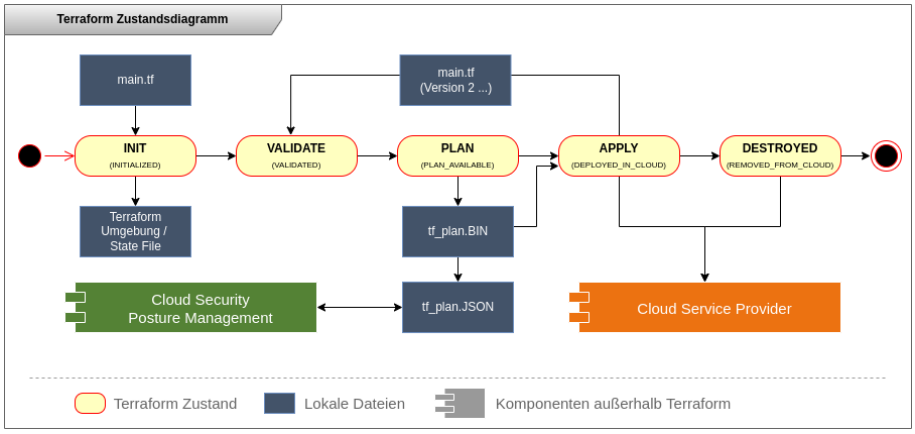

Bild 31: Terraform Zustandsmodell mit Bezug zu CSPM und der Cloudumgebung

In der einfachsten Konstellation wird die Cloud-Infrastruktur in einer Datei mit dem Namen `main.tf` entwickelt, die in komplexeren Umgebung aus mehreren dateibasierten Untermodulen bestehen kann. Der erste Terraform-Schritt dient der Initialisierung (INIT) der Umgebung und erzeugt unter anderem die Zustandsdatei (englisch state file), auf Basis der `main.tf`. Die syntaktisch Korrektheit kann über den zweiten (optionalen) Schritt der Validierung (VALIDATE) überprüft werden. Sollen keine Fehler auftreten, kann nun der Terraform-Plan (PLAN) durchgeführt und erstellt werden. Dieser Plan enthält alle Änderungen, die auf Basis der aktuellen Version der `main.tf` im Vergleich zum letzten Zustand (state file), auszuführen sind. Dieser Plan kann als binäre Datei (`tf_plan.BIN`) gespeichert werden und dient als Eingabe für den nächsten Schritt. Bisher wurden keine Änderungen in der Cloud-Umgebung ausgeführt, lediglich in den Terraform IaC-Dateien. Erst durch den Schritt APPLY wird nun der Terraform-Plan zu Ausführung gebracht und die Cloud-Umgebung entsprechend geändert.

Der CSPM-Ansatz greift nun direkt vor der Durchführung der Änderungen in der Cloud-Umgebung und prüft die Plan-Datei gegen definierte Sicherheitsregeln. Die binäre Plan-Datei muss dazu vorher in ein lesbares JSON-Format gewandelt werden.

In Listing A.3 ist eine typische Abfolge von Terraform-Befehlen dargestellt.

```
# Initialisierung, wobei implizit die Datei main.tf genutzt wird
terraform init

# Validierung der Konfiguration (main.tf, ggf. weitere Untermodule)
terraform validate

# Erstellung des Plans
# Ausgabe des Plans und Konvertierung in das JSON Format
terraform plan -out tf_plan.BIN
terraform show -json tf_plan.BIN > tf_plan.JSON
jq . tf_plan.JSON > tf_plan_pretty.JSON  # pretty-printer unter Linux

# Cloud-Infrastruktur erstellen und wieder löschen
terraform apply tf_plan.BIN
terraform destroy
```

Listing A.3: Terraform Befehlsfolge zur Erstellung und Änderung der Cloud-Umgebung

A.5.2 Terraform IaC Datenmodell für den Prototyp

Die Cloud-Infrastruktur für die prototypische Umsetzung in AWS wird auf Basis des in Listing A.4 darstellten Terraform-Code erstellt.

```
terraform {
  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 4.56.0"
    }
  }
  required_version = ">= 1.2.0"
}

provider "aws" {
  region  = "eu-central-1"
}

# ----------------------------------------------------------------
# Virtual Private Cloud und Subnetze
# ----------------------------------------------------------------

resource "aws_vpc" "MT_VPC" {
  cidr_block = "10.100.0.0/16"
  enable_dns_support = "true"   # default
  tags = {
    Name = "MT Gollwitzer VPC"
  }
}

resource "aws_subnet" "MT-Subnet-WebApps" {
  cidr_block        = "10.100.10.0/24"
  vpc_id            = aws_vpc.MT_VPC.id
  availability_zone = "eu-central-1a"
  map_public_ip_on_launch = "true"  # public network
  tags = {
    Name = "MT Gollwitzer Subnet WebApps"
  }
}

resource "aws_subnet" "MT-Subnet-BackendApps" {
  cidr_block        = "10.100.20.0/24"
  vpc_id            = aws_vpc.MT_VPC.id
  availability_zone = "eu-central-1a"
  tags = {
    Name = "MT Gollwitzer Subnet BackendApps"
  }
}

resource "aws_subnet" "MT-Subnet-Admin" {
  cidr_block        = "10.100.30.0/24"
  vpc_id            = aws_vpc.MT_VPC.id
  availability_zone = "eu-central-1a"
  map_public_ip_on_launch = "true"  # public network
```

```
  tags = {
    Name = "MT Gollwitzer Subnet Admin JumpHosts - Bastian host"
  }
}

resource "aws_vpc" "MT_VPC_2" {
  cidr_block = "10.110.0.0/16"
  enable_dns_support = "true"   # default
  tags = {
    Name = "MT Gollwitzer VPC Nr 2"
  }
}

resource "aws_subnet" "MT-Subnet-OnlyOne" {
  cidr_block        = "10.110.10.0/24"
  vpc_id            = aws_vpc.MT_VPC_2.id
  availability_zone = "eu-central-1a"
  map_public_ip_on_launch = "true"  # public network
  tags = {
    Name = "MT Gollwitzer Subnet Only One - no segmented into 2"
  }
}

# ---------------------------------------------------------------
# VPC - Gateways und Routing
# ---------------------------------------------------------------

resource "aws_internet_gateway" "MT_Internetgateway" {
  vpc_id = aws_vpc.MT_VPC.id

  tags = {
    Name = "MT InternetGateway"
  }
}

# Routing Tabelle für öffentliche Routen
resource "aws_route_table" "MT_PublicRoutetable" {
  vpc_id = aws_vpc.MT_VPC.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.MT_Internetgateway.id
  }

  tags = {
    Name = "MT Gollwitzer PublicRoutetable"
  }
}

# Zuweisung Route Table für öffentlichen Zugriff
resource "aws_route_table_association" "MT_routeTableAssociationPublicRoute" {
  route_table_id = aws_route_table.MT_PublicRoutetable.id
  subnet_id      = aws_subnet.MT-Subnet-WebApps.id
}

# ---------------------------------------------------------------
```

```
# VPC - Sicherheit Security Groups
# ----------------------------------------------------------------

resource "aws_security_group" "MT_erlaube_HTTPS" {
  name        = "mt_erlaube_tls"
  description = "Erlaube HTTPS/TLS eingehenden Datenverkehr"
  vpc_id      = aws_vpc.MT_VPC.id

  ingress {
    description      = "TLS von jeglicher Quelle"
    from_port        = 443  # Portbereich Anfang
    to_port          = 443  # Portbereich Ende
    protocol         = "tcp"
    cidr_blocks      = ["0.0.0.0/0"]
    ipv6_cidr_blocks = ["::/0"]
  }

  egress {
    from_port        = 0
    to_port          = 0
    protocol         = "-1"
    cidr_blocks      = ["0.0.0.0/0"]
    ipv6_cidr_blocks = ["::/0"]
  }

  tags = {
    Name = "mt_erlaube_tls"
  }
}

resource "aws_security_group" "MT_erlaube_HTTP" {
  name        = "mt_erlaube_http"
  description = "Erlaube HTTP unverschlüsselt"
  vpc_id      = aws_vpc.MT_VPC.id

  ingress {
    description      = "HTTP von jeglicher Quelle"
    from_port        = 80  # Portbereich Anfang
    to_port          = 80  # Portbereich Ende
    protocol         = "tcp"
    cidr_blocks      = ["0.0.0.0/0"]
    ipv6_cidr_blocks = ["::/0"]
  }

  egress {
    from_port        = 0
    to_port          = 0
    protocol         = "-1"
    cidr_blocks      = ["0.0.0.0/0"]
    ipv6_cidr_blocks = ["::/0"]
  }

  tags = {
    Name = "mt_erlaube_tls"
  }
}
```

```
resource "aws_security_group" "MT_allow_SSH_to_BastionHost" {
  name        = "mt_erlaube_ssh_to_bastion"
  description = "Erlaube eingehenden SSH-Datenverkehr aus dem Internet zu dem Bastion /
    Jump Host"
  vpc_id      = aws_vpc.MT_VPC.id

  ingress {
    description      = "SSH von jeglicher Quelle"
    from_port        = 22  # Portbereich Anfang
    to_port          = 22  # Portbereich Ende
    protocol         = "tcp"
    cidr_blocks      = ["0.0.0.0/0"] # definiert den erlaubten eingehenden IP-Bereich
    ipv6_cidr_blocks = ["::/0"]
  }

  egress {
    from_port        = 0
    to_port          = 0
    protocol         = "-1"
    cidr_blocks      = ["0.0.0.0/0"] # definiert, den Ziel-IP Bereich des ausgehenden
    Datenverkehrs
    ipv6_cidr_blocks = ["::/0"]
  }

  tags = {
    Name = "mt_erlaube_ssh_to_bastion"
  }
}

resource "aws_security_group" "MT_allow_SSH_from_BastionHost" {
  name        = "mt_erlaube_ssh_from_bastion"
  description = "Erlaube eingehenden SSH-Datenverkehr vom Bastion / Jump Host (und nicht
     direkt aus dem Internet)"
  vpc_id      = aws_vpc.MT_VPC.id

  ingress {
    description      = "SSH von Bastion Host"
    from_port        = 22  # Portbereich Anfang
    to_port          = 22  # Portbereich Ende
    protocol         = "tcp"
    cidr_blocks      = [aws_subnet.MT-Subnet-Admin.id] # definiert den erlaubten
    eingehenden IP-Bereich - nur von Bastion Host Subnet
    # ipv6_cidr_blocks = ["::/0"]
  }

  egress {
    from_port        = 0
    to_port          = 0
    protocol         = "-1"
    cidr_blocks      = [aws_subnet.MT-Subnet-Admin.id] # definiert, den Ziel-IP Bereich
    des ausgehenden Datenverkehrs
    # ipv6_cidr_blocks = ["::/0"]
  }

  tags = {
```

```
    Name = "mt_erlaube_ssh_to_bastion"
  }
}

# ---------------------------------------------------------------
# EC2 Compute Ressourcen
# ---------------------------------------------------------------

resource "aws_instance" "MT-AppServer1" {
  ami           = "ami-02365213de5bfcabf" # AMI Alpine Linux alpine-3.17.2-x86_64-bios-
    cloudinit-r0 in der Region eu-central-1
  availability_zone = "eu-central-1a"
  # iam_instance_profile
  # user_data
  instance_type = "t2.micro"
  key_name = "kp-masterthesis"      # kp-masterthesis
  # subnet_id = "subnet-0d2cee68c7d0f6893"  # subnet backoffice in VPC vpc-08
    e694d479e484ef1
  subnet_id = aws_subnet.MT-Subnet-WebApps.id
  # Zuweisung einer Security Group
  vpc_security_group_ids = [aws_security_group.MT_allow_SSH_from_BastionHost.id,
    aws_security_group.MT_erlaube_HTTPS.id, aws_security_group.MT_erlaube_HTTP.id]
  # monitoring = true #
  user_data = "Unwichtige Dinge"
  tags = {
    Name = "MT-Gollwitzer - Server 1 - Frontend"
  }
  # instance_state       = "stopped"
}

resource "aws_instance" "MT-AppServerBackend1" {
  ami           = "ami-02365213de5bfcabf" # AMI Alpine Linux alpine-3.17.2-x86_64-bios-
    cloudinit-r0 in der Region eu-central-1
  availability_zone = "eu-central-1a"
  # iam_instance_profile
  # user_data
  instance_type = "t2.micro"
  key_name = "kp-masterthesis"      # kp-masterthesis
  # subnet_id = "subnet-0d2cee68c7d0f6893"  # subnet backoffice in VPC vpc-08
    e694d479e484ef1
  subnet_id = aws_subnet.MT-Subnet-BackendApps.id
  vpc_security_group_ids = [aws_security_group.MT_allow_SSH_from_BastionHost.id]
  # monitoring = true #
  user_data = "DB password: letMEin"
  tags = {
    Name = "MT-Gollwitzer - Server 2 - Backend"
  }
  # instance_state       = "stopped"
}

resource "aws_instance" "MT-BastionHost" {
  ami           = "ami-02365213de5bfcabf" # AMI Alpine Linux alpine-3.17.2-x86_64-bios-
    cloudinit-r0 in der Region eu-central-1
  availability_zone = "eu-central-1a"
  # iam_instance_profile
  # user_data
```

```
  instance_type = "t2.micro"
  key_name = "kp-masterthesis"      # kp-masterthesis
  # subnet_id = "subnet-0d2cee68c7d0f6893"  # subnet backoffice in VPC vpc-08
    e694d479e484ef1
  subnet_id = aws_subnet.MT-Subnet-Admin.id
  # Zuweisung einer Security Group
  vpc_security_group_ids = [aws_security_group.MT_allow_SSH_to_BastionHost.id]
  # monitoring = true #
  tags = {
    Name = "MT-Gollwitzer - Bastion Host"
    IsBastionHost = "true"
  }
  # instance_state       = "stopped"
}

# ---------------------------------------------------------------
# Identity und Access Management
# ---------------------------------------------------------------

resource "aws_iam_user" "testuser1" {
  name = "testuser1"
}

resource "aws_iam_virtual_mfa_device" "secadminmfa" {
   virtual_mfa_device_name = "secadminmfa"
   # arn: arn:aws:iam::126164056430:mfa/secadmin
}


resource "aws_iam_virtual_mfa_device" "mfsandOTP" {
   virtual_mfa_device_name = "andOTP"
   # arn: arn:aws:iam::126164056430:mfa/andOTP
   # secret: ODTT3OKAV3AMDEWCTH2TR4A7P4HH4HJCTCBUBELYS7EC5NB43U66SADD4IG3ILEU
}

# ---------------------------------------------------------------
# Datenspeicher (EBS)
# ---------------------------------------------------------------

resource "aws_ebs_volume" "ebsvolume_contracts" {
  availability_zone = "eu-central-1"
  size              = 1

  tags = {
    Name = "Verträge mit Kunden aus der EU"
  }
}

resource "aws_ebs_volume" "ebsvolume-marketing" {
  availability_zone = "eu-central-1"
  size              = 1
  encrypted = true
  tags = {
    Name = "Öffentliches Marketingmaterial"
  }
}
```

```
# ---------------------------------------------------------------
# Datenspeicher Netzwerk (EFS)
# ---------------------------------------------------------------

resource "aws_efs_file_system" "EFS_Vorstand" {
  availability_zone_name = "eu-central-1"
  encrypted = true
  tags = {
    Name = "Vorstandsvorlagen"
  }
}

resource "aws_efs_file_system" "EFS_Personalakten" {
  availability_zone_name = "eu-central-1"
  tags = {
    Name = "Personalakten"
  }
}

resource "aws_efs_mount_target" "EFS_Zugriff_1" {
  file_system_id = aws_efs_file_system.EFS_Vorstand.id
  subnet_id      = aws_subnet.MT-Subnet-BackendApps.id
}

# ---------------------------------------------------------------
# Objektspeicher (S3)
# ---------------------------------------------------------------

resource "aws_s3_bucket" "bucket_rnd_drawings" {
  bucket = "RnD_bucket_CAD"

  tags = {
    Name        = "Bucket for secret Research and Development (RnD) CAD drawings of new
    products"
  }
  # Inline configuration (deprecated)
  server_side_encryption_configuration {
    rule{
      apply_server_side_encryption_by_default {
        sse_algorithm = "AES256"
      }
    }
  }
}

resource "aws_s3_bucket" "bucket_technical_manuals" {
  bucket = "bucket-technical_manuals"

  tags = {
    Name        = "Bucket for public available technical documentation and manuals for
    our products. Read for all ok."
  }
}

# ---- Policy zur Absicherung der Transportverschlüsselung bei S3 Buckets.
```

```
# Information on Terraform and AWS Policies:
# https://developer.hashicorp.com/terraform/tutorials/aws/aws-iam-policy
# Terraform API Doc: https://registry.terraform.io/providers/-/aws/4.62.0/docs/resources
    /iam_policy

# Die Policy nutzt das Listing 4.5 aus der Masterthesis (Kapitel 4.2.3 Datenspeicher ->
    Objektspeicher)
resource "aws_iam_policy" "policy_s3_https_only" {
  name        = "policy_s3_https_only"
  path        = "/"
  description = "Policy um nur transportverschlüsselten Transport zu erlauben für S3
    Buckets"

  # Die Funktion 'jsonencode' konveriert den TF Ausdruck zu einer gültigen JSON Sytntax.
  policy = jsonencode({
    Version = "2012-10-17"
    Statement = [
      {
        Action = [
          "s3:*",
        ]
        Effect   = "Deny"
        Resource = ["arn:aws:s3:::*"] # gilt für alle S3 buckets
        Condition = {
          Bool = {
            "aws:SecureTransport" = "false"
          }
        }
      },
    ]
  })
}

# ---- Konfiguration zur serverseitigen Verschlüsselung der S3 Buckets
resource "aws_s3_bucket_server_side_encryption_configuration" "s3_serverside_encryption"
     {
  bucket = aws_s3_bucket.bucket_rnd_drawings.id

  rule {
    apply_server_side_encryption_by_default {
      # kms_master_key_id = aws_kms_key.mykey.arn # optional argument, when using own
    keys.
      sse_algorithm     = "aws:kms"
    }
  }
}
# resource "aws_s3_bucket_server_side_encryption_configuration" "
    s3_serverside_encryption2" {
#   bucket = aws_s3_bucket.bucket_technical_manuals.id

#   rule {
#     apply_server_side_encryption_by_default {
#       # kms_master_key_id = aws_kms_key.mykey.arn # optional argument, when using own
    keys.
#       sse_algorithm     = "aws:kms"
#     }
```

```
#   }
# }

# resource "aws_iam_user_ssh_key" "userkey1" {
#   username   = aws_iam_user.tfuser1.name
#   encoding   = "SSH"
#   public_key = "ssh-rsa
    AAAAB3NzaC1yc2EAAAADAQABAAABAQD3F6tyPEFEzV0LX3X8BsXdMsQz1x2cEikKDEY0aIj41qgxMCP/
    iteneqXSIFZBp5vizPvaoIR3Um9xK7PGoW8giupGn+EPuxIA4cDM4vzOqOkiMPhz5XK0whEjkVzTo4+
    S0puvDZuwIsdiW9mxhJc7tgBNL0cYlWSYVkz4G/fslNfRPW5mYAM49f4fhtxPb5ok4Q2Lg9dPKVHO/
    Bgeu5woMc7RY0p1ej6D4CKFE6lymSDJpW0YHX/wqE9+
    cfEauh7xZcG0q9t2ta6F6fmX0agvpFyZo8aFbXeUBr7osSCJNgvavWbM/06
    niWr0vYX2xwWdhXmXSrbX8ZbabVohBK41 mytest@mydomain.com"
# }
```

Listing A.4: Terraform basierte AWS Infrastruktur

A.6 Open Policy Agent - Prototypische Umsetzung CSPM

A.6.1 Open Policy Agent Laufzeitumgebung

Open Policy Agent auf der Kommandozeile

Die einfachste Form mit der OPA-Laufzeitumgebung zu interagieren, ist die Ausführung über die Kommandozeile. Das ausführbare Programm `opa` bietet mit den Optionen `exec` und `eval` zwei Arten die Policies gegen einen zu prüfenden Datensatz auszuführen. Das Listing A.5 zeigt anhand eines einfachen Beispiels die Ausführung. Im Verzeichnis `policy` sind alle REGO-Dateien abgelegt. Die Datei `terraform_plan.json` beinhaltet die zu prüfende Cloud-Konfiguration in Form des Terraform Plans, formatiert als JSON. Ausgeführt wird nur die Policy `policy_aws_1` aus dem Package `wings/mt-gollwitzer/aws`.

```
# Ausführen einer Policy, aus einem Policy-Bundle, gegen den  Terraform-Plan
opa eval "data.wings.mtgollwitzer.aws.policy_aws_2" --bundle ./policy/ --input ./
    terraform_plan.json
```

Listing A.5: Open Policy Agent - Kommandozeile Beispiel mit eval

Open Policy Agent als REST-Server

Für produktive Umgebungen bietet es sich an den OPA im Server-Modus zu starten. Der Server-Modus bietet eine REST-API zu Kommunikation mit dem Dienst an. Der Dienst arbeitet im Standard auf Port 8181 und über das HTTP-Protokoll. Der Dienst ist konfigurierbar und kann auch Transportverschlüsselung über TLS anbieten. Die REST-API Operationen dienen dazu Sicherheitsregeln (REGO Policies) zu landen oder zu aktualisieren, Daten in den Server zu laden, oder Abfragen zur Prüfung von JSON-formatierten Daten durchzuführen.

Das Bild 32 veranschaulicht den Server-Modus. Der OPA-Dienst kann bereits mit einem einfachen Kommandozeilen-Programm wie cURL (Client for URLs) genutzt werden, da alle REST-API Operationen über HTTP arbeiten können. Beim Start des OPA-Diensts können die Sicherheitsregeln bereits geladen werden, indem diese in eine sogenannte Bundle-Datei zusammengeführt werden. Der Client (rechts im Bild) nutzt das Kommandozeilen-Programm cURL und sendet die zu prüfende Cloud-Konfiguration in Form der Terraform Plan-Datei (JSON formatiert) an den

OPA-Dienst. Dieser prüft die Cloud-Konfiguration nun automatisiert gegen die Policies aus dem Bundle und gibt bei Sicherheitsfehlern eine JSON-formatierte Rückmeldung.

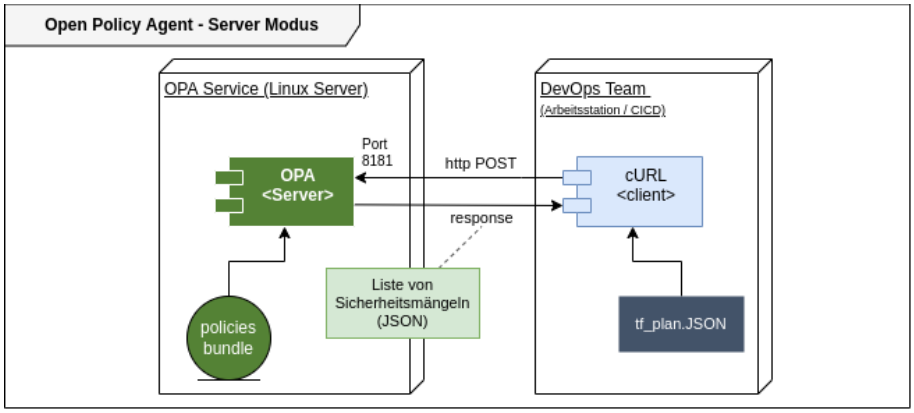

Bild 32: Open Policy Agent - Server Modus mit einfachem Linux-Client

```
# 1. Erzeugen des Policy-Bundle auf Basis der REGO-Policy Dateien im VZ policy
opa build ./policy

# 2. Starten des OPA-Servers, laden der Policies im Bundle
opa run --server -b bundle.tar.gz

# 3. Client: Aufruf des Servers zur Prüfung der Cloud-Konfiguration
#   3.1 vorab: wandeln der Terraform-Datei in ein input-JSON Konstrukt
cat <<EOF > v1-data-tf_plan.json
{
    "input": $(cat tf_plan.JSON)
}
EOF

#   3.2 Aufruf des OPA-Dienstes (und pretty print JSON Rückgabe)
curl localhost:8181/v1/data/wings/mt-gollwitzer/policy_aws_1 -d @v1-data-tf_plan.json -H
     'Content-Type: application/json' | json_pp

# 2. b) Alternativ kann die Prüfung auch auf der Kommandozeile im Modus EXEC erfolgen
opa exec -b bundle.tar.gz terraform_plan.json
```

Listing A.6: Open Policy Agent - Server und Client Interaktion

A.6.2 Programm zur Ergebnisaufbereitung

Das Programm in Listing A.7 steuert den Aufruf des OPA-Dienstes. Es werden die Policies (als bundle Datei) übergeben, sowie die zu prüfende Terraform Plan Datei.

```
import json
```

```
import subprocess
from prettytable import PrettyTable
import datetime

# Execution syntax
# opa exec -b bundle.tar.gz terraform_plan.json > evaluation-response.json

newline = "\n"

def main():

  now = datetime.datetime.now()
  # Aufruf der OPA Engine mit Uebergabe des Policy-Bundles, sowie des aktuellen
    Tearraform PLAN als JSON-Datei
  opa_result = subprocess.run(["opa", "exec", "-b", "bundle.tar.gz", "terraform_plan.
    json"], capture_output=True, text=True)
  # Rueckgabe aus OPA-Engine zu JSON Objekt konvertieren
  opa_reponse = opa_result.stdout
  data = json.loads(opa_reponse)
  # JSON Daten analysieren und strukturiert auf die Konsole ausgeben
  inputFileProcessed = data["result"][0]["path"]
  sResultToConsoleHeader = "
    ==========================================================================================
    " + newline
  sResultToConsoleHeader += " ANALYSEBERICHT Cloud Security Posture Management | Version
     1.0 | MT Andreas Gollwitzer" + newline
  sResultToConsoleHeader += newline
  sResultToConsoleHeader += "  Datum:\t\t" + now.strftime("%d/%m/%Y, %H:%M:%S") +
    newline
  sResultToConsoleHeader += "  Terraform Plan:\t" + inputFileProcessed + newline
  sResultToConsoleHeader += "  Policy Bundle:\t" +  "bundle.tar.gz" + newline
  sResultToConsoleDetails = ""
  # Prüfe ob das Gesamtergebnis erlaubt oder unerlaubt ist.
  allowed = data["result"][0]["result"]["allow"]
  if not allowed:
    counterHigh = 0
    counterMedium = 0
    counterLow = 0
    violations = data["result"][0]["result"]["violations"]
    sResultToConsoleHeader += "  Anzahl Fehler:\t" + str(len(violations)) + newline
    counter = 0
    for violation in violations:
      counter += 1
      FehlerhafteRessource = violation[1]["Misconfigure Ressources"]
      Sicherheitsmangel = violation[2]["Sicherheitsmangel"]
      Title = violation[0]["Title"]
      Kritikalitaet = violation[0]["Kritikalitaet"]
      if Kritikalitaet == "HIGH":
        counterHigh += 1
      elif Kritikalitaet == "MEDIUM":
        counterMedium += 1
      elif Kritikalitaet == "LOW":
        counterLow += 1
      Beschreibung = violation[0]["Beschreibung"]
      Sicherheitsempfehlung = violation[0]["Sicherheitsempfehlung"]
      Behebung = violation[0]["Behebung"]
```

```
      BehebungCLI = violation[0]["Behebung-CLI"]
      Compliance = violation[0]["Compliance"]
      Referenzen = violation[0]["Referenzen"]
      sResultToConsoleDetails += "
    ----------------------------------------------------------------------------------------------------
    " + newline
      sResultToConsoleDetails += "  Sicherheitsmangel [" + str(counter) + "/" + str(len(
    violations)) + "]: " + Sicherheitsmangel + newline + newline
      # sResultToConsoleDetails += "  Fehlerhafte Ressourcen:" + FehlerhafteRessource +
    newline
      sResultToConsoleDetails += "  " + "Policy:\t\t\t" + Title + newline
      sResultToConsoleDetails += "  " + "Kritikalitaet:\t\t" + Kritikalitaet  + newline
      sResultToConsoleDetails += "  " + "Beschreibung:\t\t\t" + Beschreibung + newline
      sResultToConsoleDetails += "  " + "Sicherheitsempfehlung:\t" +
    Sicherheitsempfehlung + newline
      sResultToConsoleDetails += "  " + "Behebung:\t\t\t" + Behebung + newline
      sResultToConsoleDetails += "  " + "Compliance:\t\t\t" + Compliance + newline
      for referenz in Referenzen:
        sResultToConsoleDetails += "  " + "Referenz:\t\t\t" + referenz["description"] +
    "\n\t\t\t\t" + referenz["ref"] + newline
        # print(json.dumps(data, indent = 4, sort_keys=True))

  sResultToConsoleHeader += "    Severity HIGH:\t" + str(counterHigh) + newline
  sResultToConsoleHeader += "    Severity MEDIUM:\t" + str(counterMedium) + newline
  sResultToConsoleHeader += "    Severity LOW:\t" + str(counterLow) + newline
  sResultToConsoleHeader += "
    ==================================================================================================
    " + newline
  sResultToConsole = sResultToConsoleHeader
  sResultToConsole += sResultToConsoleDetails
  sResultToConsole += "
    ==================================================================================================
    " + newline
  print(sResultToConsole)

if __name__ == "__main__":
    main()
```

Listing A.7: CSPM Programm zum Aufruf der OPA-Engine und Ergebnisaufbereitung

A.6.3 Strukturvorlage für OPA-basierte CSPM-Policies

Die in Kapitel 3.3.2 entwickelte fachliche Struktur der Sicherheitsrichtlinie (siehe Bild 6) ist in ein technisches und maschinell lesbares Format umzusetzen. In Listing A.8 ist die technische Umsetzung der fachlichen Vorgaben auf Kapitel 3.3.2 dargestellt. Die Umsetzung nutzt eine Kombination aus vordefinierten, als auch ergänzenden und selbstdefinierten Annotationen.

```
# METADATA
# scope:package
```

```
# title: Masterthesis Andreas Gollwitzer - Prototypische Implementierung CSPM Policies
    auf Basis REGO
# description: Eine Auswahl an CSPM-Regeln für die Cloudumgebung AWS, auf Basis der
    erarbeiteten Best-Practises für Cloud Security Posture Management
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# organizations:
# - Hochschule Wismar, Fakultät für Ingenieurwissenschaften, Bereich Elektrotechnik und
    Informatik
package wings.mtgollwitzer.aws

# ------------------------------------
# Bezug zur formalen Defition aus Kapitel 3.3.2 Verwaltung von Sicherheitsrichtlinien (
    Bild 5 - Fachliche Struktur einer Sicherheitsrichtlinie für Policy-as-Code)
# ID -> METADATA.title
# Titel -> METADATA.title
# Kritikalität -> METADATA.custom.severity
# Beschreibung -> METADATA.description
# Sicherheitsempfehlung -> METADATA.custom.advice
# Prüflogik -> REGO basierter Code
# Behebung -> METADATA.custom.remediation und .remediationautomation
# Compliance -> METADATA.custom.compliance
# Referenzen -> METADATA.related_resources
# ------------------------------------
# siehe https://www.openpolicyagent.org/docs/latest/policy-language/#annotations
# ------------------------------------

# Strukturvorlage für CSPM Policies

# METADATA
# scope: rule
# title: <ID> und sprechender Titel
# description: Fachliche Beschreibung welche Regel(n) auf welche Cloud-Ressourcen
    angewendet werden.
# related_resources: Liste von URLs die beispielsweise auf Quellen für diese CSPM-Regel
    darstellen.
# - ref: https://avd.aquasec.com/misconfig/aws/ec2/default-security-group/
# - ref: https://gsl.dome9.com/D9.AWS.NET.78.html
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: MEDIUM
#  advice: Beschreibung welche Cloud-Konfiguration empfehlenswert ist bzw. welche zu
    vermeiden oder untersagt ist.
#  remediation: Beschreibung der Schritte zur Fehlerkorrektur.
#  remediationautomation: Command Line Interface Code zur Behebung des
    Konfigurationsfehler (optional, wenn sinnvoll)
#  compliance: Verweise auf relevante Kontrollen der Sicherheitsstandards. Trennung
    durch Komma.
policy_aws_metadata[results] {

    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
```

```
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die VPCs %s weisen keine Segmentierung auf und verletzen somit die
    Regel '%s'", [erroneous_ressources, annotation.title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources":
    erroneous_ressources}, decision}

}
```

Listing A.8: Technische Umsetzung der beschreibenden Informationen zur Policy auf Basis der METADATA Struktur in OPA

A.6.4 OPA Policies

In diesem Kapitel sind die Quellcodedateien für die Open-Policy-Agent (OPA) basierte prototypische Umsetzung dokumentiert. Der Quellcode ist in der deklarativen Regel-Sprache Rego programmiert, die sich an der Programmiersprache beziehungsweise Abfragesprache Datalog orientiert [49].

Das Listing A.9 stellt in einer OPA-Ausführungsumgebung den Standard-Einstieg dar mit dem Paket `system` und der Regel `main`. Alle im Rahmen der Masterthesis entwickelten Regeln, für die AWS basierte Infrastruktur, wurden im Paket `wings.mtgollwitzer.aws` entwickelt. Regeln werden in OPA in den `data` Namensraum geladen, daher ist der Präfix `data` notwendig. Alle Regeln wurden gleichnamig entwickelt unter dem Namen `awspolicy`, dies kann als eine übliche Vorgehensweise in OPA angesehen werden und ermöglicht somit die Prüfung welche und wie viele der Regeln nicht erfolgreich absolviert wurden.

```
package system

import future.keywords
import data.wings.mtgollwitzer.aws as mtawspolicies

main := {
  "allow": count(mtawspolicies.awspolicy) == 0,
  "violations": mtawspolicies.awspolicy,
}
```

Listing A.9: OPA Policy: System Main als oberste Ebene

AWS Policy 1

```
package wings.mtgollwitzer.aws

import future.keywords.if
import future.keywords.in

# -------------------------------
# Policy
# -------------------------------


# METADATA
# scope: rule
# title: AWS1 - Use subnets to isolate the tiers of your application within a single VPC
    .
# description: Das VPC Netzwerk ist ein einzelne Subnetze zu gliedern. Die EC2 Instanzen
     sind diesen Subnetzen zuzuordnen. EC2 Instanzen dürfen nicht außerhalb von
    Subnetzen existieren.
# related_resources:
# - ref: https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/infrastructure-security.
    html
#   description: "AWS Security - Infrastructury Security"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: MEDIUM
#  advice: Prüfen Sie alle EC2 Instanzen (aws_instance), ob diese einem Subnetz
    zugeordnet sind. Stellen Sie sicher, dass nur EC2 Instanzen die öffentlich
    erreichbar sein müssen in public Subnetzen sind. Alle anderen EC2 Instanzen sind
    privaten Subnetzen zuzuordnen.
#  remediation: Die EC2 Instanzen (TF Ressource aws_instance) sind mittels dem Parameter
     'subnet_id' einem Subnetz zuzuordnen.
#  remediationautomation: AWS CLI 'aws ec2 stop-instances --instance_ids ' oder
    Terraform 'resource "aws_ec2_instance_state" {state = "stopped"}
#  compliance: CIS 12.2, CIS 12.8
awspolicy[results]{
    non_segmented_vpcs := check_vpcs_for_segmentation
    count(non_segmented_vpcs) > 0
    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die VPCs %s weisen keine Segmentierung auf und verletzen somit die
    Regel '%s'", [non_segmented_vpcs, annotation.title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources":
    non_segmented_vpcs}, decision}
}

# Prüft ob alle VPC 2 oder mehr Subnetze aufweisen
```

```
# Rückgabe: result ist eine Liste alles VPC-Namen, die nicht der Bedingung entsprechen 2
     oder mehr Subnetze aufzuweisen.
check_vpcs_for_segmentation[result] {
    vpcs := all_vpcs
    some vpc in vpcs
      mysubnets := subnets_by_vpc[vpc]
      # wenn die Anzahl der Subnetzte < 2 ist, wird das VPC der Fehlerliste hinzugefügt
      count(mysubnets) < 2
      result := vpc
}

# Erzeugt eine Liste aller VPCs mit den zugeordneten Subnetzen
# Technisch wird ein Objekt erzeugt mit Key = VPC-Name und den Werten als Array Subnetz-
    Name(n)
subnets_by_vpc[vpc] := subnets {
    # alle VPCs suchen
    resources := input.resource_changes[_]
    resources.type == "aws_vpc"
    vpc := resources.name
    # vpcs := [vpc | vpc := resources.name]
    # suche alle Subnetze die dem VPC zugeordnet sind (in der TF-plan Sektion '
    configuration')
    subnets := [subnet | # Query / suche / selektiere alle Ressourcen, iteriere über
    alle Ressourcen
                         resources := input.configuration.root_module.resources[_]
                         # Aus der Liste aller Ressourcen nehme nur die Subnetze
                         resources.type == "aws_subnet"
                         # Aus der Liste alle Subnetze nehme nur die in der Referenz
    eine Beziehung zur VPC haben
                         resources.expressions.vpc_id.references[_] == concat("", ["
    aws_vpc.", vpc])
                         # Setze den Namen des Subnetzes in die Ergebnisliste (das Array
    )
                         subnet := resources.name
                ]
}

# Liste aller VPCs
all_vpcs[vpc] {
   vpcs := input.resource_changes[_]
   vpcs.type == "aws_vpc"
   vpc := vpcs.name
}

# Liste aller Subnetze
all_subnets[subnet] {
   # resources := input.configuration.root_module.resources[_]
   resources := input.resource_changes[_]
   resources.type == "aws_subnet"
   subnet := resources.name
}
```

Listing A.10: OPA Policy: AWS 1

AWS Policy 2

```
package wings.mtgollwitzer.aws
import future.keywords.if

# METADATA
# scope: rule
# title: AWS2 - Restrict access to your instances using security groups.
# description: Der Zurgiff auf EC2 Instanzen sollte stets durch eine Firewall (Security
    Group) geschützt werden. Die Security Gruppe sollte den Zugriff auf die notwendige
    Kommunikation beschränken (eigehende Ports, IP-Bereiche einschränken etc).
# related_resources:
# - ref: https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/infrastructure-security.
    html
#   description: "AWS Security - Infrastructury Security"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: HIGH
#  advice: Prüfen Sie alle EC2 Instanzen (aws_instance), ob diese einer Security Group
    zugeordnet sind.
#  remediation: Die EC2 Instanzen (TF Ressource aws_instance) sind mittels dem Parameter
     'vpc_security_group_ids' einer Security Group zugeordnet.
#  remediationautomation: AWS CLI 'aws ec2 stop-instances --instance_ids ' oder
    Terraform 'resource "aws_ec2_instance_state" {state = "stopped"}
#  compliance: CIS 13.4, CIS 13.9, CIS 13.10
awspolicy[results]{
    instances := instances_without_security_group
    count(instances) > 0
    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die EC2 Instanz(en) %s verletzen die Regel '%s'", [instances,
    annotation.title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources": instances},
    decision}
}

instances_without_security_group[instancelist] {
   all_instances := all_aws_instances
   # Iteriere über alle EC2 Instanzen
   instance := all_instances[_]
   # Selektiere nur die EC2 Instanzen, die keiner Security Group zugeordnet sind (die
    kein Attribut vpc_security_group_ids aufweisen)
   not instance.expressions.vpc_security_group_ids
   # Die EC2 Instanzen ohne SG werden der Liste der fehlerhaften EC2 Instanzen
    hinzugefügt.
```

```
   instancelist := instance.name
}

all_aws_instances[instancelist] {
   # instances := input.resource_changes[_]
   instances := input.configuration.root_module.resources[_]
   instances.type == "aws_instance"
   instancelist := instances
}

# ----------------------------------------------
# Dokumentation zur Regellogik:
# ----------------------------------------------
# Eine Security Group wird in der TF-Datei einer EC2 Instanz über das Attribut
    vpc_security_group_ids zugewiesen
# Beispiel: vpc_security_group_ids = [aws_security_group.my_security_group.id]

# Im TF-Plan wird die Security Group unter folgendem JSON-Pfad deklatiert:
# configuration.root_module.resources[_].type = "aws_instance"
# configuration.root_module.resources[_].expressions.vpc_security_group_ids.references[]
# wenn dieser JSON Knoten nicht vorhanden ist, ist die EC2 Instanz keiner SG zugeordnet.

# Die Details (ein / ausgehende Regeln) der SG können im TF-Plan unter dem JSON-Pfad
    ausgelesen werden:
# configuration.root_module.resources[_].type = "aws_security_group"
#   Name der SG: .name = <Name>
#   Regeln: .expressions.egress und .expressions.ingress
#   -> constant_value[].cidr_blocks[]
#   -> constant_value[].from_port
#   -> constant_value[].to_port
#   -> constant_value[].protocol ("tcp", "udp", "icmp")
```

Listing A.11: OPA Policy: AWS 2

AWS Policy 3

```
package wings.mtgollwitzer.aws
import future.keywords.if

# METADATA
# scope: rule
# title: AWS3 - Use a bastion host or NAT gateway for internet access from an instance
    in a private subnet.
# description: Der administrative Zugriff auf EC2 Instanzen innerhalb einer privaten
    Subnetzes darf nicht direkt erfolgen. Der Zugriff muss stets über einen sogenannten
    Bastion Host erfolgen. Nur der Bastion Host ist im öffentlichen Netz, oder über ein
    VPN, von außen erreichbar.
# related_resources:
# - ref: https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/infrastructure-security.
    html
#   description: "AWS Security - Infrastructury Security"
# - ref: https://aws.amazon.com/de/solutions/implementations/linux-bastion/
#   description: "Referenzbereitstellung eines Bastion Host 'as a service' auf AWS"
```

```
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: HIGH
#  advice: Alle EC2 Instanzen in privaten Subnetzten müssen über eine Security Group
    geschützt werden, die nur Zugriff von der IP des Bastion Host auf den SSH Port 22
    erlaubt.
#  remediation: Die EC2 Instanzen (TF Ressource 'aws_instance') sind mittels dem
    Parameter 'vpc_security_group_ids' einer Security Group zugeordnet. Die Security
    Group (Ressource 'aws_security_group') muss unter 'expressions.ingress' im CIDR
    Block nur die IP des Bastion Host aufweisen. Der Bastion Host ist per Tag
    gekennzeichnet.
#  remediationautomation: AWS CLI 'aws ec2 stop-instances --instance_ids ' oder
    Terraform 'resource "aws_ec2_instance_state" {state = "stopped"}
#  compliance: CIS 13.4, CIS 13.9, CIS 13.10
awspolicy[results]{
    # Ermittle die EC2 Instanz die als Bastion Host typisiert ist (Tag IsBastionHost)
    # bastionhost := getbastionhost
    bastionhost := input.configuration.root_module.resources[_]
    bastionhost.type == "aws_instance"
    bastionhost.expressions.tags.constant_value["IsBastionHost"] == "true"
    # Lese die SubnetID des Bastion Host aus
    bastionhostsubnet := bastionhost.expressions.subnet_id.references[0]
    # Suche die Security Group die als Quelle und Ziel das Subnetz des Bastion-Host hat
    und nur Port 22 eingehend erlaubt
    sg_allowfrombastion := input.configuration.root_module.resources[_]
    sg_allowfrombastion.type == "aws_security_group"
    sg_allowfrombastion.expressions.ingress.references[0] == bastionhostsubnet
    sg_name := sg_allowfrombastion.name

    # Suche alle EC2 Instanzen, die nicht der Security Group (Bastion Host Zugriff
    erlauben) zugeordnet sind.
    instances := input.configuration.root_module.resources[_]
    instances.type == "aws_instance"
    not instances.name == bastionhost.name # Bastion host aus der Trefferliste
    ausblenden.
    not instances.expressions.vpc_security_group_ids.references[1] == concat(".",["
    aws_security_group",sg_name])
    instance_names := instances.name
    # Zusatz 1: Prüfe nur EC2 Instanzen in privaten Subnetzen sind (d.h. Subnetz hat
    nicht map_public_ip_on_launch = true)
    # public_subnets := input.configuration.root_module.resources[_]
    # public_subnets.type == "aws_subnet"
    # public_subnets.expressions.map_public_ip_on_launch
    # public_subnets.expressions.map_public_ip_on_launch.constant_value != "true"

    # Zusatz 2: Prüfe, ob die zugeordneten Secruity Groups einen direkten Zugriff per
    SSH erlauben. Nur diese Instanzen sind gefähret.
    # assigned_security_groups := instances.expressions.vpc_security_group_ids.
    references[_]
    # assigned_security_groups.expressions.ingress.to_port == 22

    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
```

```
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die EC2 Instanz(en) '%s' verletzen die Regel '%s'", [instance_names,
     annotation.title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigured Ressources": instance_names
    }, decision}
}

# Sucht die EC2 Instanzen, die den Tag "IsBastionHost" auf "true" gesetzt haben.
getbastionhost[result]{
    bastionhost := input.configuration.root_module.resources[_]
    bastionhost.type == "aws_instance"
    bastionhost.expressions.tags.constant_value["IsBastionHost"] == "true"
    result := bastionhost
}

get_sg_allowfrombastion[sg_name]{
    sg_allowfrombastion := input.configuration.root_module.resources[_]
    sg_allowfrombastion.type == "aws_security_group"
    sg_allowfrombastion.expressions.ingress.references[1] == "aws_subnet.MT-Subnet-Admin
    "
    sg_name := sg_allowfrombastion.name
}

# ----------------------------------------------
# Dokumentation zur Regellogik:
# ----------------------------------------------
# Die Regellogik prüft derzeit auf das administrative Subnetz, in dem der Bastion Host
    installiert ist.
# Eine verbesserte Logik könnte zusätzlich prüfen, ob das Network interface des Bastion
    Host definiert ist (als eigene Ressource) und dessen ID in der Secuirty Group als
    Regel definiert ist.
```

Listing A.12: OPA Policy: AWS 3

AWS Policy 4

```
package wings.mtgollwitzer.aws
import future.keywords.if

# METADATA
# scope: rule
# title: AWS4 - Verwenden Sie für jedes Konto die Multi-Factor Authentication (MFA).
# description: Alle AWS Benutzer müssen das Multi-Factor Authentifizierungsverfahren
    einsetzen und dazu mindestens ein MFA Gerät aktiviert haben.
# related_resources:
# - ref: https://docs.aws.amazon.com/de_de/IAM/latest/UserGuide/data-protection.html
#   description: "AWS Security - Datenschutz in AWS Identity and Access Management"
```

```
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: HIGH
#  advice: Prüfen Sie alle IAM Benutzer, ob für den Benutzer ein MFA-Gerät (virtual
    Device) zugeordnet ist.
#  remediation: Benutzerzugänge ohne MFA sind temporär zu sperren und es ist zB per AWS
    CLI ein MFA Gerät hinzuzufügen.
#  remediationautomation: AWS CLI 'aws iam enable-mfa-device --user-name --
    authentication-code1' oder Terraform 'resource "aws_iam_virtual_mfa_device"' nutzen.
#  compliance: CIS 14.3
awspolicy[results]{
    # Terraform kennt zwar die Ressourcern aws_iam_user und aws_iam_virtual_mfa_device,
    es konnte jedoch nach einem Import der Cloud-Konfiguration
    # mittels des Kommandos 'terraform import ...' keine Relation zwischen dem User und
    dem MFA Endgerät im TF-Plan erkannt werden.
    # Im TF-Plan sind beide Ressourcen ersichtlich, habe aber eine Referenz / Relation.
    Daher kann auch keine Regel programmiert werden.
    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := "Regel konnte nicht implementiert werden, der Bezug zwischen IAM User und MFA
     Device ist nicht im TF-Plan erkennbar."
    results := {{"Sicherheitsmangel": msg}, decision}
}

# ----------------------------------------------
# Dokumentation zur Regellogik:
# ----------------------------------------------
# Terraform kennt zwar die Ressourcern aws_iam_user und aws_iam_virtual_mfa_device, es
    konnte jedoch nach einem Import der Cloud-Konfiguration
# mittels des Kommandos 'terraform import ...' keine Relation zwischen dem User und dem
    MFA Endgerät im TF-Plan erkannt werden.
# Im TF-Plan sind beide Ressourcen ersichtlich, haben aber keine Referenz / Relation
    zueinander. Daher kann auch keine Regel programmiert werden, die bereits
# vor einem Deployment die Gültigkeit prüfen könnte.
#
# Relevante Ressourcen in Terraform:
# aws_iam_user
# aws_iam_virtual_mfa_device
```

Listing A.13: OPA Policy: AWS 4

AWS Policy 5

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in

# METADATA
# scope: rule
# title: AWS5 - Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and
    recommend TLS 1.3.
# description: ergänzen
# related_resources:
# - ref: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/data-protection.html
#   description: "AWS Security - Data protection in Amazon EC2"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: HIGH
#  advice: Die HTTP-basierte Kommunikation zu AWS Ressourcen (zB Diensten auf EC2)
    sollten stets transportverschlüsselt erfolgen. Der Einsatz von TLS wird empfohlen.
#  remediation: Unverschlüsselte HTTP-Kommunikation ist beispielsweise durch
    vorgelagerte Firewalls (Security Group) zu unterbinden und die Service Komponente (
    zB der Webserver) ist entsprechend zu konfigurieren (siehe Referenz).
#  remediationautomation: nicht automatisierbar - die TLS Konfiguration muss auf dem
    WebServer erfolgen.
#  compliance: CIS 3.10
awspolicy[results]{
    # Prüfe die Security Groups der EC2 Instanze, ob diese den Port 80 eingehend
    erlauben.
    instances := input.configuration.root_module.resources[_]
    instances.type == "aws_instance"
    # Finde Security Grouppen, die einen Zugriff auf Port 80 zulassen. Diese sind nicht
    erwünscht.
    unwanted_sg := sg_with_80_ingress
    # Iteriere über alle der EC2 Instanz zugeordneten Security Groups und prüfe ob die
    ungewollte SG zugeordnet ist.
    some reference in instances.expressions.vpc_security_group_ids.references
      # die Variable beinhaltet die Referenz auf die Security Gruppe(n) der EC2-Instanz.
     Wenn diese in dem Set (Liste) unwanted vorhanden ist, gibt der Ausdruck true zurück
    . Treffer.
      unwanted_sg[reference]
    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die EC2 Instanz(en) ''%s' verletzen die Regel '%s'. Die Security-
    Group erlaubt Zugriff auf dem unsicheren Port 80.", [instances.name, annotation.
    title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources": instances.name},
     decision}
}
```

```
# Prüft, ob Security Gruppen existieren, die den eigehenden Datenverkehr auf Port 80
    erlauben.
sg_with_80_ingress[sgs]{
  check_for_port := 80
  security_groups := input.configuration.root_module.resources[_]
  security_groups.type == "aws_security_group"
  security_groups.expressions.ingress.constant_value[0].from_port <= check_for_port
  security_groups.expressions.ingress.constant_value[0].to_port >= check_for_port
  sgs := concat(".", ["aws_security_group", security_groups.name])
}

public_subnets[subnets]{
  public_subnets := input.configuration.root_module.resources[_]
  public_subnets.type == "aws_subnet"
  # Selektiere nur Subnetze, die öffentliche sind (öffentliche IPs vergeben)
  public_subnets.expressions.map_public_ip_on_launch
  subnets := public_subnets.name
}
```

Listing A.14: OPA Policy: AWS 5

AWS Policy 6

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in

# METADATA
# scope: rule
# title: AWS6 - Use AWS encryption solutions [...] Encryption at rest for your EBS
    volumes and snapshots.
# description: Die Verschlüsselung der Daten im Ruhezustand stellt ein weiteres
    Sicherheitsmerkmal dar. Die Sicherheitsregel verlangt bei Speicherlösungen, wie AWS
    EBS, die Verschlüsselung encryption-at-rest aktiviert zu haben.
# related_resources:
# - ref: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/data-protection.html
#   description: "AWS Security - Data protection in Amazon EC2"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: MEDIUM
#  advice: Es ist zu prüfen, dass alle EBS Volumes die Option zur Verschlüsselung im
    Ruhezustand aktiviert haben. Dies kann nur bei der Erstellung des EBS Volumen
    erfolgen, nachträglich ist dies nicht mehr möglich.
#  remediation: Unverschlüssewlte EBS-Speicher müssen auf einen verschlüsselten (neuen)
    EBS-Spcher kopiert werden. Dies ist ein weitgehend manueller Arbeitsvorgang.
#  remediationautomation: nicht sinnvoll automatisierbar.
#  compliance: CIS 3.11
awspolicy[results]{
    ebsvolumes := input.configuration.root_module.resources[_]
    ebsvolumes.type == "aws_ebs_volume"
    # prüft ob in der Konfiguration des EBS Volume die Verschlüsselung data-at-rest
    aktiviert ist. Wenn nicht, ist dies ein Regelbruch.
```

```
    not ebsvolumes.expressions.encrypted
    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die EBS Volumen(en) '%s' bzw. Snapshots '%s' verletzen die Regel '%s
    '. Die Verschlüsselung data-at-rest ist zu aktivieren.", [ebsvolumes.name,
    check_encrpytion_ebs_snapshots, annotation.title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources": ebsvolumes.name
    }, decision}
}

check_encrpytion_ebs_snapshots[notencrpyted] {
   ebssnapshots := input.configuration.root_module.resources[_]
   ebssnapshots.type == "aws_ebs_snapshot"
   notencrpyted := ebssnapshots.name
}
```

Listing A.15: OPA Policy: AWS 6

AWS Policy 7

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in

# Namen erlaubter SSH Schlüssel
allowed_ssh_keys := {"kp-masterthesis", "zweiterSSHKeyNamen"}

# METADATA
# scope: rule
# title: AWS8 - Use AWS encryption solutions [...] Remote access encryption using SSH
# description: SSH provides a secure communications channel for remote access to your
    Linux instances.
# related_resources:
# - ref: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AccessingInstancesLinux.
    html
#   description: "Connect to your Linux instance using SSH"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: MEDIUM
#  advice: Administrative Zugriffe sind durch den Einsatz kryptografischer
    Verschlüsselung zu sichern. Die administrativen Zugänge dürfen keine
    unverschlüsselte Kommunikation nutzen. Es ist das Secure Socket Shell (SSH)
    Protokoll zu nutzen.
```

```
#  remediation: Administrationsanleitung siehe Referenzen.
#  remediationautomation: Nicht automatisiert möglich.
#  compliance: CIS 3.11
awspolicy[results]{
    instances := input.configuration.root_module.resources[_]
    instances.type == "aws_instance"
    # Der kryptografische Schlüssel für den SSH Zugang wird unter dem Attribut key_name
    übergeben. Dieser muss existieren
    not instances.expressions.key_name.constant_value

    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die EC2 Instanz '%s' verletzen die Regel '%s'. Es ist kein Schlüssel
     für die SSH Kommunikation konfiguriert.", [instances.name, annotation.title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources": instances.name},
     decision}
}

#-------------------------------------------------------------------------------
# Ausbauoptionen:
# 1) Alternativ kann auch gegen eine Liste erlaubter Schlüssel geprüft werden (
    allerdings nur auf Basis des Names)
```

Listing A.16: OPA Policy: AWS 7

AWS Policy 8

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in

# METADATA
# scope: rule
# title: AWS8 - Enable At Rest Encryption for Elastic File System (EFS)
# description: If your organization is subject to corporate or regulatory policies that
    require encryption of data and metadata at rest, we recommend creating a file system
     that is encrypted at rest, and mounting your file system using encryption of data
    in transit.
# related_resources:
# - ref: https://avd.aquasec.com/misconfig/aws/efs/avd-aws-0037/
#   description: "EFS Encryption has not been enabled"
# - ref: https://docs.aws.amazon.com/efs/latest/ug/encryption.html
#   description: "Data encryption in Amazon EFS"
# - ref: https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/
    efs_file_system
```

```
#   description: "Terraform Resource: aws_efs_file_system"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: MEDIUM
#  advice: Sensitive Daten sind durch den Einsatz kryptografischer Verschlüsselung zu
    sichern. Es ist mindestens die Encryption-at-Rest Funktion von AWS zu nutzen. Des
    weiteren können die Dateien selbst durch ein client-seitiges
    Verschlüsselungsverfahren zusätzlich geschützt werden, sowie mittels einer geeigente
     Härtung der Berechtigungen ein erhähtes Schutzniveau erreicht werden.
#  remediation: Eine nachträgliche Verschlüsselung eines bestehenden EFS
    Netzwerkspeichers wird derzeit nicht unterstützt. Es ist ein neues verschlüsseltes
    EFS anzulegen und die Daten sind zu kopieren. Aktivierung der Verschlüsselung siehe
    angegebene Referenz.
#  remediationautomation: Nicht automatisiert möglich.
#  compliance: CIS 3.11
awspolicy[results]{
    efsvolumes := input.configuration.root_module.resources[_]
    efsvolumes.type == "aws_efs_file_system"

    not efsvolumes.expressions.encrypted
    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Der/die EFS Netzwerkspeicher '%s' verletzen die Regel '%s'. Die
    Verschlüsselung data-at-rest ist zu aktivieren.", [efsvolumes.name, annotation.title
    ])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources": efsvolumes.name
    }, decision}
}

#-------------------------------------------------------------------------------
# Ausbauoptionen:
# 1) Eine Analyse des zugehörige EFS Mount Target könnte erfolgen, um zB zu prüfen, in
    welchem
#    Netzwerk (öffentlich, privat) dieses liegt.
# 2) Es könnte auch analysiert werden, ob dem Mount Target eine Security Group
    zugeordnet wurde und diese SG
#    den Datenverkehr auf bestimmte Quell-IP Bereiche einschränkt.
#    Die OPA Logik kann basierend auf Regel AWS2 erfolgen (anstatt EC2 als Basis-
    Ressource nun EFS Ressourcen verwenden).
```

Listing A.17: OPA Policy: AWS 8

AWS Policy 9

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in


# ----------------
# list of words not allowed in user_data (exmaple, list can be extended)

insecure_words := {"password", "username", "secret", "credentials", "user", "passwd"}

# METADATA
# scope: rule
# title: AWS9 - No Secrets In User Data
# description: EC2 instance data is used to pass start up information into the EC2
    instance. This userdata must not contain access key credentials. Instead use an IAM
    Instance Profile assigned to the instance to grant access to other AWS Services.
# related_resources:
# - ref: https://avd.aquasec.com/misconfig/aws/ec2/avd-aws-0029/
#   description: "AWS EC2 - No Secrets In User Data "
# - ref: https://docs.aws.amazon.com/efs/latest/ug/encryption.html
#   description: "Data encryption in Amazon EFS"
# - ref: https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/
    efs_file_system
#   description: "Terraform Resource: aws_efs_file_system"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: MEDIUM
#  advice: siehe auch Masterthesis, Kapitel 4.2.2 Rechendienste und Ressourcen,
    Abschnitt Instanz-Metadaten.
#  remediation: Eine nachträgliche Verschlüsselung eines bestehenden EFS
    Netzwerkspeichers wird derzeit nicht unterstützt. Es ist ein neues verschlüsseltes
    EFS anzulegen und die Daten sind zu kopieren. Aktivierung der Verschlüsselung siehe
    angegebene Referenz.
#  remediationautomation: Nicht automatisiert möglich.
#  compliance: CIS 3.11
awspolicy[results]{
    instances := input.configuration.root_module.resources[_]
    instances.type == "aws_instance"
    # Selektiere / filtere nur Instanzen, die user_data konfiguriert haben.
    instances.expressions.user_data
    # Wenn user_data vorhanden ist, wird der Inhalt der Daten untersucht (vor Deployment
    , danach ist ein Zugriff auf die User-Daten nur aus der EC2 Instanz möglich)
    user_data := instances.expressions.user_data.constant_value
    # Der Wert darf bestimmte Zeichenketten nicht enthalten. Es wird über die Liste (Set
    ) insecure_words geschleift, ob eines der Worte vorhanden ist
    # Wenn ein Wort vorhanden ist, wird die Instanz gefiltert.
    some word in insecure_words
      contains(user_data, word)

    # Ausbauoption: Prüfe, ob die Instanz zusätzlich kein "iam_instance_profile"
    referenziert / nutzt
    # not instances.expressions.iam_instance_profile
    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
```

```
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die EC2 Instanz(en) '%s' verletzen die Regel '%s'. Die Metadaten
    UserData dürfen keine sensitiven Daten enthalten.", [instances.name, annotation.
    title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources": instances.name},
     decision}
}

#-----------------------------------------------------------------------------
# Ausbauoptionen:
# 1) Die Inhalte des User-Data könnten auch mittels forensischer Ansätze untersucht
    werden, beispielsweise die Entropie einzelner Wörter zu ermitteln
# 2) oder mittels einer Passwortliste zu untersuchen (Wörterbuchangriff / dictionary
    attack) - zB Quelle: https://www.kali.org/tools/wordlists/
```

Listing A.18: OPA Policy: AWS 9

AWS Policy 10

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in

# METADATA
# scope: rule
# title: AWS10 - S3 Secure Transport Enabled
# description: All statements in all S3 bucket policies must have a condition that
    requires encryption at a certain level. S3 buckets support numerous types of
    encryption, including AES-256, KMS using a default key, KMS with a CMK, or via HSM-
    based key.
# related_resources:
# - ref: https://avd.aquasec.com/misconfig/aws/s3/s3-bucket-encryption-enforcement/
#   description: "S3 Bucket Encryption Enforcement"
# - ref: https://docs.aws.amazon.com/AmazonS3/latest/userguide/UsingServerSideEncryption
    .html
#   description: "Protecting data using server-side encryption with Amazon S3 managed
    encryption keys"
# - ref: https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/
    s3_bucket_server_side_encryption_configuration
#   description: "Terraform API Doc - Provides a S3 bucket server-side encryption
    configuration resource."
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: HIGH
#  advice: siehe Masterthesis, Kapitel 4.2.3 Datenspeicher, Abschnitt Objektspeicher.
    Sowie Tabelle 14 zur AWS CLI zur Prüfung.
```

```
#  remediation: Configure a bucket policy to enforce encryption.
#  remediationautomation: encryption must be enabled when S3 bucket is created.
#  compliance: CIS 3.11
awspolicy[results]{
     # Liefert ein Set-Objekt von Buckets die data-at-rest verschlüsselt sind.
     encryptedBuckets := getAllS3BucketsEncryptedServerside
     # Alle S3 buckets in der Konfiguration
     allBuckets := getAllS3BucketsByName
     is_set(allBuckets)
     is_set(encryptedBuckets)
     # Prüft, welche Buckets _nicht_ in der Liste der verschlüsselten Bucktets sind.
     # check if a bucket is not allready in the list of encrypted buckets
     some bucket in allBuckets
       not encryptedBuckets[bucket]
       nonEncryptedBuckets := bucket

    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }

    msg := sprintf("Die S3 Buckets '%s' verletzen die Regel '%s'. Die S3 Buckets müssen
    server-seitige im Ruhezustand verschlüsselt sein.", [nonEncryptedBuckets, annotation
    .title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources":
    nonEncryptedBuckets}, decision}
}

getAllS3BucketsEncryptedServerside[encryptedBuckets]{
    # instances := input.configuration.root_module.resources[_]
    # instances.type == "aws_s3_bucket"
    s3_encrpyption_configs := input.configuration.root_module.resources[_]
    s3_encrpyption_configs.type == "aws_s3_bucket_server_side_encryption_configuration"
    # 1. Prüft ob eine der Regeln die Verschlüsselung der Daten im Ruhezustand festlegt.
    #    "apply_server_side_encryption_by_default" muss dazu gesetzt sein.
    s3_encrpyption_configs.expressions.rule[_].apply_server_side_encryption_by_default
    # Liefert ein Array der referenzierten S3 Buckets
    some bucketReferences in s3_encrpyption_configs.expressions.bucket.references
      not endswith(bucketReferences, ".id")
      encryptedBuckets := bucketReferences
}

# queries for all S3 buckets and returen their name with prefix "aws_s3_bucket."
getAllS3BucketsByName[buckets] {
    s3buckets := input.configuration.root_module.resources[_]
    s3buckets.type == "aws_s3_bucket"
    buckets := concat(".", {"aws_s3_bucket", s3buckets.name})
}
```

Listing A.19: OPA Policy: AWS 9

AWS Policy 11

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in

# METADATA
# scope: rule
# title: AWS11 - S3 Secure Transport Enabled
# description: Ensure AWS S3 buckets enforce SSL to secure data in transit. S3 buckets
    should be configured to strictly require SSL connections.
# related_resources:
# - ref: https://avd.aquasec.com/misconfig/aws/s3/s3-secure-transport-enabled/
#   description: "S3 Secure Transport Enabled"
# - ref: https://repost.aws/knowledge-center/s3-bucket-policy-for-config-rule
#   description: "What S3 bucket policy should I use to comply with the AWS Config rule
    s3-bucket-ssl-requests-only?"
# - ref: https://registry.terraform.io/providers/-/aws/4.67.0/docs/resources/iam_policy
#   description: "Terraform API Doc - Resource: aws_iam_policy"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: HIGH
#  advice: siehe Masterthesis, Kapitel 4.2.3 Datenspeicher, Abschnitt Objektspeicher.
    Die Kommunikation zu S3 Buckets ist stets mittel Transportverschlüsselung
    abzusichern.
#  remediation: Es ist eine IAM Policy zu erstellen (siehe Referenzen) und diese ist auf
     alle S3 Ressourcen anzuwenden.
#  remediationautomation:
#  compliance: CIS 3.10
awspolicy[results]{
    not s3_TLS_enforcement_policy_existent

    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := "Es existiert keine IAM Policy, die für alle S3 Buckets eine HTTPS-
    Transportverschlüsselung erzwingt."
    results := {{"Sicherheitsmangel": msg}, decision}
}
```

```
s3_TLS_enforcement_policy_existent{
  instances := input.planned_values.root_module.resources[_]

  # Es wird eine IAM Policy gesucht, die auf alle S3 Ressourcen wirkt und HTTP-
    Transportverschlüsselung erzwingt.
  instances.type == "aws_iam_policy"

    # Die Policy ist in Terraform in einem JSON-Objekt enkodiert. Zur Analyse muss dies
    zunächst unmarshalled werden.
    # Es wird damit ein Rego-Objekt erzeugt, das ein Rego Array "Statement" enthält.
    unmarshalledPolicy := json.unmarshal(instances.values.policy)
    # 1. Bedingung ist, dass die Policy sich auf S3 Buckets (Ressourcen) beziehen muss,
    sowie auf Aktion mit S3 Buckets
    ressourcen := unmarshalledPolicy.Statement[0].Resource[_]
    contains(ressourcen, "aws:s3")
    actions := unmarshalledPolicy.Statement[0].Action[_]
    contains(actions, "s3")

    # 2. Die Policy muss eine Kondition enthalten, die auf einen SecureTransport ==
    false prüft.
    # Die Kondition wird als Rego Objekt übergeben, somit als unsortiertes Schlüssel-
    Wert Paar (key value pair)
    kondition := unmarshalledPolicy.Statement[0].Condition.Bool
    kondition["aws:SecureTransport"] == "false"

    # 3. Bedingung ist, dass dies nun unterbunden werden muss.
    unmarshalledPolicy.Statement[0].Effect == "Deny"
}
```

Listing A.20: OPA Policy: AWS 11

AWS Policy 12

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in

# METADATA
# scope: rule
# title: AWS12 - S3 Secure Transport Enabled
# description: Ensure AWS S3 buckets enforce SSL to secure data in transit. S3 buckets
    should be configured to strictly require SSL connections.
# related_resources:
# - ref: https://avd.aquasec.com/misconfig/aws/s3/s3-secure-transport-enabled/
#   description: "S3 Secure Transport Enabled"
# - ref: https://repost.aws/knowledge-center/s3-bucket-policy-for-config-rule
#   description: "What S3 bucket policy should I use to comply with the AWS Config rule
    s3-bucket-ssl-requests-only?"
# - ref: https://registry.terraform.io/providers/-/aws/4.67.0/docs/resources/iam_policy
#   description: "Terraform API Doc - Resource: aws_iam_policy"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
```

```
#  severity: HIGH
#  advice: siehe Masterthesis, Kapitel 4.2.3 Datenspeicher, Abschnitt Objektspeicher.
    Die Kommunikation zu S3 Buckets ist stets mittel Transportverschlüsselung
    abzusichern.
#  remediation: Es ist eine IAM Policy zu erstellen (siehe Referenzen) und diese ist auf
     alle S3 Ressourcen anzuwenden.
#  remediationautomation:
#  compliance: CIS 3.10
awspolicy[results]{
    not s3_TLS_enforcement_policy_existent

    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := "Es existiert keine IAM Policy, die für alle S3 Buckets eine HTTPS-
    Transportverschlüsselung erzwingt."
    results := {{"Sicherheitsmangel": msg}, decision}
}


s3_TLS_enforcement_policy_existent{
  instances := input.planned_values.root_module.resources[_]

  # Es wird eine IAM Policy gesucht, die auf alle S3 Ressourcen wirkt und HTTP-
    Transportverschlüsselung erzwingt.
  instances.type == "aws_iam_policy"

    # Die Policy ist in Terraform in einem JSON-Objekt enkodiert. Zur Analyse muss dies
    zunächst unmarshalled werden.
    # Es wird damit ein Rego-Objekt erzeugt, das ein Rego Array "Statement" enthält.
    unmarshalledPolicy := json.unmarshal(instances.values.policy)
    # 1. Bedingung ist, dass die Policy sich auf S3 Buckets (Ressourcen) beziehen muss,
    sowie auf Aktion mit S3 Buckets
    ressourcen := unmarshalledPolicy.Statement[0].Resource[_]
    contains(ressourcen, "aws:s3")
    actions := unmarshalledPolicy.Statement[0].Action[_]
    contains(actions, "s3")

    # 2. Die Policy muss eine Kondition enthalten, die auf einen SecureTransport ==
    false prüft.
    # Die Kondition wird als Rego Objekt übergeben, somit als unsortiertes Schlüssel-
    Wert Paar (key value pair)
    kondition := unmarshalledPolicy.Statement[0].Condition.Bool
    kondition["aws:SecureTransport"] == "false"

    # 3. Bedingung ist, dass dies nun unterbunden werden muss.
    unmarshalledPolicy.Statement[0].Effect == "Deny"
}
```

Listing A.21: OPA Policy: AWS 12

AWS Policy 13

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in

# Basis: CIS Amazon Web Services Foundations Benchmark (v1.5.0)

# METADATA
# scope: rule
# title: AWS13 - Ensure the default security group of every VPC restricts all traffic
# description: A VPC comes with a default security group whose initial settings deny all
     inbound traffic, allow all outbound traffic, and allow all traffic between
    instances assigned to the security group. If you don't specify a security group when
     you launch an instance, the instance is automatically assigned to this default
    security group. Security groups provide stateful filtering of ingress/egress network
     traffic to AWS resources. It is recommended that the default security group
    restrict all traffic.
# related_resources:
# - ref: https://workbench.cisecurity.org/sections/844441/recommendations/1387898
#   description: "CIS Amazon Web Services Foundations Benchmark (v1.5.0) - rule 5.4"
# - ref: https://docs.aws.amazon.com/cli/latest/reference/ec2/describe-security-groups.
    html
#   description: "AWS Command Line Interface (CLI) reference - describe-security-groups"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: HIGH
#  advice: Configuring all VPC default security groups to restrict all traffic will
    encourage least privilege security group development and mindful placement of AWS
    resources into security groups which will in-turn reduce the exposure of those
    resources.
#  remediation: see reference from CIS, section Remediation Procedure.
#  remediationautomation: Remove any inbound and outbound rules from named Security
    Groups.
#  compliance: CIS 3.3

awspolicyManual[results]{
    # Default Security Gruppen werden automatisch von AWS _nach dem_ erstellen einer VPC
     erzeugt.
    # Die hier erstellten Regel greifen _vor dem_ Deployment.
    # Eine Regel die nach dem Deployment analysieren würde, welche Konfiguration die
    Default Security Gruppe der VPC hat, würde wie folgt gestaltet sein.
    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
```

```
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die Default Security Group(s) der genannten VPCs '%s' sind gegen die
     Regel '%s' manuell, mittels AWS CLI, zu prüfen. Siehe Referenz, dazu ist das
    Kommando aws ec2 describe-security-groups --group-ids <ID der SG> zu nutzen.", [
    getDefaultSecurityGroupsByVPC, annotation.title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources":
    getDefaultSecurityGroupsByVPC}, decision}
}

getDefaultSecurityGroupsByVPC[results]{
    # Default Security Gruppen werden automatisch von AWS _nach dem_ erstellen einer VPC
     erzeugt.
    # Die hier erstellten Regel greifen _vor dem_ Deployment.
    # Eine Regel die nach dem Deployment analysieren würde, welche Konfiguration die
    Default Security Gruppe der VPC hat, würde wie folgt gestaltet sein.
    vpcs := input.resource_changes[_]
    vpcs.type == "aws_vpc"
    defaultSecurityGroupId := vpcs.change.after.default_security_group_id
    # Terraform zeigt in der Plan Datei die erstellte Security Group ID dar, aber nicht
    die Konfiguration der Security Group.
    # Die Analyse muss daher auf Basis des Command Line Interfaces (ausserhalb OPA)
    erfolgen.
    # AWS CLI: aws ec2 describe-security-groups --group-ids <ID der SG>
    #   Prüfern der Regeln für eingehende Daten:
    #     ip-permission.cidr / ip-permission.from-port und ip-permission.to-port / ip-
    permission.protocol
    #   Prüfen der Regeln für ausgehende Daten:
    #     egress.ip-permission.cidr / egress.ip-permission.from-port und egress.ip-
    permission.to-port / egress.ip-permission.protocol
    results := {vpcs.name, defaultSecurityGroupId}
}
```

Listing A.22: OPA Policy: AWS 13

AWS Policy 14

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in

# Basis: CIS Amazon Web Services Foundations Benchmark (v1.5.0)

# METADATA
# scope: rule
# title: AWS14 - Ensure the default security group of every VPC restricts all traffic
# description: A VPC comes with a default security group whose initial settings deny all
     inbound traffic, allow all outbound traffic, and allow all traffic between
    instances assigned to the security group. If you don't specify a security group when
     you launch an instance, the instance is automatically assigned to this default
    security group. Security groups provide stateful filtering of ingress/egress network
```

```
     traffic to AWS resources. It is recommended that the default security group
    restrict all traffic.
# related_resources:
# - ref: https://workbench.cisecurity.org/sections/844441/recommendations/1387898
#   description: "CIS Amazon Web Services Foundations Benchmark (v1.5.0) - rule 5.4"
# - ref: https://docs.aws.amazon.com/cli/latest/reference/ec2/describe-security-groups.
    html
#   description: "AWS Command Line Interface (CLI) reference - describe-security-groups"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: HIGH
#  advice: Configuring all VPC default security groups to restrict all traffic will
    encourage least privilege security group development and mindful placement of AWS
    resources into security groups which will in-turn reduce the exposure of those
    resources.
#  remediation: see reference from CIS, section Remediation Procedure.
#  remediationautomation: Remove any inbound and outbound rules from named Security
    Groups.
#  compliance: CIS 3.3

awspolicyManual[results]{
    # Default Security Gruppen werden automatisch von AWS _nach dem_ erstellen einer VPC
     erzeugt.
    # Die hier erstellten Regel greifen _vor dem_ Deployment.
    # Eine Regel die nach dem Deployment analysieren würde, welche Konfiguration die
    Default Security Gruppe der VPC hat, würde wie folgt gestaltet sein.
    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die Default Security Group(s) der genannten VPCs '%s' sind gegen die
     Regel '%s' manuell, mittels AWS CLI, zu prüfen. Siehe Referenz, dazu ist das
    Kommando aws ec2 describe-security-groups --group-ids <ID der SG> zu nutzen.", [
    getDefaultSecurityGroupsByVPC, annotation.title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources":
    getDefaultSecurityGroupsByVPC}, decision}
}

getDefaultSecurityGroupsByVPC[results]{
    # Default Security Gruppen werden automatisch von AWS _nach dem_ erstellen einer VPC
     erzeugt.
    # Die hier erstellten Regel greifen _vor dem_ Deployment.
    # Eine Regel die nach dem Deployment analysieren würde, welche Konfiguration die
    Default Security Gruppe der VPC hat, würde wie folgt gestaltet sein.
    vpcs := input.resource_changes[_]
    vpcs.type == "aws_vpc"
    defaultSecurityGroupId := vpcs.change.after.default_security_group_id
```

```
    # Terraform zeigt in der Plan Datei die erstellte Security Group ID dar, aber nicht
    die Konfiguration der Security Group.
    # Die Analyse muss daher auf Basis des Command Line Interfaces (ausserhalb OPA)
    erfolgen.
    # AWS CLI: aws ec2 describe-security-groups --group-ids <ID der SG>
    #   Prüfern der Regeln für eingehende Daten:
    #     ip-permission.cidr / ip-permission.from-port und ip-permission.to-port / ip-
    permission.protocol
    #   Prüfen der Regeln für ausgehende Daten:
    #     egress.ip-permission.cidr / egress.ip-permission.from-port und egress.ip-
    permission.to-port / egress.ip-permission.protocol
    results := {vpcs.name, defaultSecurityGroupId}
}
```

Listing A.23: OPA Policy: AWS 14

AWS Policy 15

```
package wings.mtgollwitzer.aws
import future.keywords.if
import future.keywords.in

# Erstellt auf Basis der erarbeiteten Tabelle 6 aus der Masterthesis
allowed_availability_zones_EU := {"eu-central-1a", "eu-west-1", "eu-west-2", "eu-west
    -3", "eu-south-1", "eu-north-1"}

# METADATA
# scope: rule
# title: AWS15 - Ensure compliance with regional regulations by using appropriate
    Regions in AWS global network
# description: Landesspezifische rechtliche Vorgaben, oder auch unternehmensspezifische
    Regeln, können die Speicherung und Verarbeitung bestimmter Daten auf regionale Zonen
     einschränken. Die regionale Lage der Daten, der EC2 Instanzen oder EBS Volumen, ist
     zu prüfen.
# related_resources:
# - ref: https://aws.amazon.com/de/compliance/gdpr-center/
#   descritpion: "DSGVO-Compliance bei der Nutzung von AWS-Services"
# - ref: https://aws.amazon.com/de/about-aws/global-infrastructure/
#   description: "Globale AWS-Infrastruktur"
# - ref: https://repost.aws/knowledge-center/move-ec2-instance
#   descritpion: "How do I move my EC2 instance to another subnet, Availability Zone, or
     VPC?"
# authors:
# - Andreas Gollwitzer <a.gollwitzer@stud.hs-wismar.de>
# custom:
#  severity: MEDIUM
#  advice: Prüfen Sie die Regionen / Verfügbarkeitszonen der AWS Ressourcen (compute und
     storage), in welcher AWS Region die Daten verarbeitet werden.
#  remediation: Die EC2 Instanz ist zu sichern (AMI erzeugen) und in einer neuen Region
    zu erzeugen, auf Basis des AMI, und zu starten.
#  remediationautomation: keine Automatisierung.
#  compliance: EU-GDPR
awspolicy[results]{
```

```
    instances := input.configuration.root_module.resources[_]
    instances.type == "aws_instance"
    # Vorgabe: alle Instanzen müssen in europäischen Rechenzentren verarbeitet werden.
    not allowed_availability_zones_EU[instances.expressions.availability_zone.
    constant_value]

    annotation := rego.metadata.rule()
    decision := {
        "Title": annotation.title,
        "Kritikalitaet": annotation.custom.severity,
        "Beschreibung": annotation.description,
        "Sicherheitsempfehlung": annotation.custom.advice,
        "Behebung": annotation.custom.remediation,
        "Behebung-CLI": annotation.custom.remediationautomation,
        "Compliance": annotation.custom.compliance,
        "Referenzen": annotation.related_resources,
    }
    msg := sprintf("Die EC2 Instanz(en) '%s' verstößt gegen die Sicherheitregel ''%s'",
    [instances.name, annotation.title])
    results := {{"Sicherheitsmangel": msg}, {"Misconfigure Ressources": instances.name},
     decision}
}

# Ausbau:
# 1) Weitere Ressourcen einbeziehen (EFS, S3); am besten in eigene Funktionen auslagern.
```

Listing A.24: OPA Policy: AWS 15

API: `EC2.DescribeInstances, DescribeInstanceStatus, DescribeInstanceAttribute`		
Pfad/Objekt	**Schlüssel**	**Werte und CSPM-Relevanz**
Reservations[]	ownerId	Die ID des AWS Account der als Eigner der Instanz(-Reservierung) geführt wird. Anmerkung: Je Instanz wird eine Reservierung angelegt. Dies wird ab hier nun zur besseren Lesbarkeit vernachlässigt. Frage(n): 2
InstancesSet[]	instanceId	Eindeutige Kennzeichnung der Instanz. Frage(n): 1
	imageId	Referenz ID des verwendeten Abbilds, Verweis auf Objekt AMI, Beispiel: `ami-02bdef6edc8a29773`. Frage(n): 4
	instanceState.code	Statuscode als 16-bit Integer. Werte (Auswahl): 16 running, 80 stopped, 48 terminated (wird gelöscht). Frage(n): 1
	placement.availabilityZone	Code(s) der Verfügbarkeitszone(n) in der die Instanz installiert wurde. Frage(n): 3
	vpcId, subnetId	Eindeutige Kennzeichnung der AWS Virtual Private Cloud, bzw. des Subnetzes, in der die Instanz angelegt wurde. Beispiel: `vpc-08e694d479e484ef1`. Frage(n): 6
	ipAddress, ipv6Address, privateIpAddress, dnsName	Öffentliche IPv4 bzw. die IPv6 Adresse. Die `privateIpAddress` stellt die interne IPv4 Adresse dar. Frage(n): 6
	groupSet[]	Liste mit jeweils `groupId` und `groupName` als Referenzen auf SecurityGroup Objekte (Beispiel: `sg-0cb0abf6d9c45c39c`). Frage(n): 10
	blockDeviceMapping[].ebs.volumeId	Eine Liste mit Block-Geräten die der Instanz zugeordnet sind. Jedes Gerät wird u.a. mit `volumeId`, `status` (attached) beschrieben. Frage(n): 7
	keyName	Fremdschlüssel zu kryptographischem Schlüsselpaar. Frage(n): 8

Tabelle 8: API Objekte - Computing Ressourcen

API: `EC2.DescribeImages` und Datentyp `Image`		
Pfad/Objekt	**Schlüssel**	**Werte und CSPM-Relevanz**
ImageSet[]	imageId	Primärschlüssel.
	name	Frei wählbarer Namen.
	description	Beschreibung welches AMI als Basis dieses AMIs diente. Frage(n): 5
	platform	Im Falle von Windows wird `windows` gesetzt, ansonsten leer. Frage(n): 5
	platformDetails	Informationen zur Abrechnung (billing).
	imageOwnerId	Die AWS Account ID des Herstellers des Images. Frage(n): 4
	imageOwnerAlias	Der AWS Account Alias (z.B. *amazon*, oder *self*), oder auch die Account ID. Frage(n): 4

Tabelle 9: API Objekt der Amazon Machine Images

API: `EC2.DescribeInstances, DescribeInstanceAttribute` und Datentyp `InstanceMetadataOptionsResponse`		
Pfad/Objekt	**Schlüssel**	**Werte und CSPM-Relevanz**
metadataOptions	httpEndpoint	Ein Schalter der bestimmt, ob der Metadaten-Zugriff für die Instanz erlaubt ist. Der Wert `disabled` unterbindet den Zugriff. Frage(n): 9 a)
	httpTokens	Wird der Wert `optional` (Default) genutzt, sind beide Versionen des IMDS zulässsig. Wird `required` gesetzt, ist nur IMDSv2 erlaubt. IMDSv2 verlangt eine Token-basierte Zugriffssteuerung. Frage(n): 9 b)
InstanceSet[]	iamInstanceProfile	Fremdschlüssel (ID, Amazon Resource Name (ARN)) für ein IAM Instanz-Profil, falls ein solches der Instanz zugeordnet wurde. Frage(n): 9 c)
userData	value	Liefert die User-Data Metadaten zurück, jedoch base64 enkodiert. Frage(n): 9 d)

Tabelle 10: API Objekt zur Instanz-Metadaten Konfiguration

API: `EC2.DescribeVolumes und DescribeSnapshots` mit Filter auf OwnerId=<eigener Account> bzw. status private		
Pfad/Objekt	**Schlüssel**	**Werte und CSPM-Relevanz**
Volumes[]	Encrypted	Wenn der Wert auf `true` steht, ist der Datenträger verschlüsselt. Frage(n): 1
	Attachments[]	Wenn diese Liste leer ist, ist der Datenträger keiner EC2 Instanz zugeordnet. Ansonsten ist die zugeordnete EC2 Instanz über den Schlüssel `InstanceId` zu identifizieren. Frage(n): 2
	AvailabilityZone	Das Feld beinhaltet den AWS Regionen-Code (siehe auch Tabelle 5 in Anhang A.2). Frage(n): 3
Snapshots[]	Encrypted	Wenn der Wert auf `true` steht, ist der Snapshot verschlüsselt. Frage(n): 4

Tabelle 11: Relevante API Objekt und Wert-Schlüssel-Paare für den Blockspeicher EBS

API: `EFS.DescribeFileSystems, DescribeMountTargets, DescribeFileSystemPolicy, DescribeMountTargetSecurityGroups und EC2.DescribeSecurityGroups`		
Pfad/Objekt	**Schlüssel**	**Werte und CSPM-Relevanz**
FileSystems[]	Encrypted	Wenn der Wert auf `true` steht, ist der Datenträger verschlüsselt. Frage(n): 1
	AvailabilityZoneName, FileSystemArn	Die Zeichenketten enthalten den AWS Regionen-Code (siehe auch Tabelle 5 in Anhang A.2). Frage(n): 3
Policy.Statement[]	aws:SecureTransport	Die Kondition lässt die Transportverschlüsselung erzwingen. Wenn der Wert *false* ist, wird der Zugriff per Effect *Deny* in der Policy verweigert. Frage(n): 2
	Principal	Bestimmt welcher AWS-IAM Benutzer oder welche Gruppe/Rolle Zugriff erhält. Frage(n): 4
	Condition	Der EFS-Speicher unterscheidet drei Actions. `ClientMount` lesender Zugriff, `ClientWrite` schreibender Zugriff und `ClientRootAccess` ermöglicht die Steuerung des root-Zugriff durch sogenanntes NFS 'root squashing', bei dem der root-user des Clients auf einen Anonymous-User auf dem EFS-Dateisystem gemapped wird. Frage(n): 4
SecurityGroups[]	IpPermissions[]	Erlaubte eingehende Verbindungen mit `FromPort, IpProtocol, IpRanges (CIDR)` Frage(n): 5
	IpPermissionsEgress[]	Regeln für den ausgehenden Datenverkehr. Frage(n): 5

Tabelle 12: Relevante API Objekt und Wert-Schlüssel-Paare für den Netzwerkspeicher EFS

API: `S3.GetBucketAcl` und `GetBucketPolicy`, `GetBucketEncryption`, `GetBucketLocation` und `GetObjectAcl`		
Pfad/Objekt	**Schlüssel**	**Werte und CSPM-Relevanz**
GetBucketEncryption	ServerSideEncryption-Configuration	Die Verschlüsselung neuer Bucket-Objekte erfolgt server-seitig automatisiert, wenn die `ApplyServerSideEncryptionByDefault` Regel gesetzt ist. Frage(n): 1
GetBucketPolicy	Condition	Damit eine Transportverschlüsselung erzwungen wird, muss in einer Bucket-Policy eine Prüfung der `aws:SecureTransport` Konfiguration erfolgen. Ist der Wert `false` muss der Zugriff verweigert werden, d.h. die Policy hat den `Effect='Deny'` zu nutzen. Frage(n): 2
GetBucketLocation	LocationConstraint	Das Feld beinhaltet den AWS Regionen-Code (siehe auch Tabelle 5 in Anhang A.2). Frage(n): 3
GetBucketAcl, GetObjectAcl	Grants	Die Zugriffe werden in die Berechtigungen `READ`, `WRITE`, `FULL_CONTROL` gegliedert. Umfangreiche Berechtigungen für die vorgefertigten AWS Benutzergruppen `AuthenticatedUsers und AllUsers` stellen in der Regel ein Sicherheitsrisiko dar und sind als Werte der Direktive `Grantee` zu prüfen. Frage(n): 4
GetBucketPolicy	Condition	AWS empfiehlt die Zugriffsberechtigungen nicht über ACL zu steuern, sondern Policies und IAM zu nutzen. Die Nutzung von ACL kann unterbunden werden, wenn Policies mit der Bedingung `'s3:x-amz-object-ownership': 'BucketOwnerEnforced'` zu Aktionen wie `s3:CreateBuckets` geprüft werden. Ist `BucketOwnerEnforced` nicht gesetzt, ist die Anfrage abzulehnen Frage(n): 6,4
GetBucketPolicy	Condition	Es sollte eine Bedingung existieren, die auf die Quell-IP prüft und einschränkt. Direktiven `IpAddress` und `aws:SourceIp` sind zu prüfen. Frage(n):5

Tabelle 13: API Objekte - Objektspeicher S3

API: `EC2.DescribeVpcs, DescribeSubnets, DescribeInternetGateways, DescribeNetworkAcls, DescribeRouteTables, DescribeSecurityGroups, DescribeSecurityGroupRules, DescribeVpnConnections, 53.GetDNSSEC, GetDomainDetail`

Pfad/Objekt	Schlüssel	Werte und CSPM-Relevanz
VpcSet[]	vpcId	Primärschlüssel.
	isDefault, state	Kann Rückschlüssel auf Standardeinstellungen geben, sowie ob die VPC aktiv ist. Frage(n): 1
SubnetSet[]	subnetId, vpcId	Primärschlüssel.
	mapPublicIpOnLaunch	Indikator ob öffentliche IPv4 automatisch den EC2 Instanzen zugewiesen werden. Frage(n): 2
networkAclSet[]	networkAclId	Primärschlüssel.
	cidrBlock	IPv4-Bereich. Frage(n): 1, 2
	portRange	Port-Bereich. Frage(n): 1, 2
	protocol	Protokollnummer oder -1 für Alle. Frage(n): 1, 2
	ruleAction	Werte: allow oder deny als Aktion der Regel. Frage(n): 1, 2
securityGroup[]	groupId	Primärschlüssel.
	ipPermissions[]	Liste von Regeln (IpPermission Objekte) für eingehenden Datenverkehr. Frage(n): 2, 5
	ipPermissionsEgress[]	Liste von Regeln für ausgehenden Datenverkehr. Frage(n): 2, 5
	formPort und toPort	Port-Bereich bei TCP/UDP, bei ICMP die Typnummer / Code. Frage(n): 2, 5
	ipProtocol	Protokollname (tcp, udp...) oder -nummer. Frage(n): 2, 5
DomainDetail	AdminPrivacy, RegistrantPrivacy, TechPrivacy	Boolscher Wert, zeigt an ob die Kontaktinformationen bei einer WHOIS-Abfrage bereitgestellt werden. Frage(n): 4
DNSSECStatus	ServeSignature	Werte SIGNING oder NOT_SIGNING. Zeigt an, ob DNSSEC aktiviert ist für die Domäne. Frage(n): 4

Tabelle 14: API Objekt virtualisiertes Netzwerk und Netzwerksicherheit

API: `kms.list-keys, describe-key,enable-key`		
Pfad/Objekt	**Schlüssel**	**Werte und CSPM-Relevanz**
Keys[]	KeyId	Primärschlüssel.
KeyMetadata	Enabled	Boolescher Wert (true, false). Frage(n): 1,3
	KeyUsage	Kryptografische Funktion des Schlüssels: ENCRYPT_DECRYPT, GENERATE_VERIFY_MAC oder SIGN_VERIFY Frage(n): 1
	EncryptionAlgorithms	Unterstütze Algorithmen zur Ver- und Entschlüsselung. Allerdings! wird bei Standard AWS Schlüsseln kein Detail ausgeben, sondern z.B. nur 'SYMMETRIC_DEFAULT' Frage(n): 2
	SigningAlgorithms	Bei Schlüsseln zur Signatur. Auch hier kann 'SYMMETRIC_DEFAULT' ausgegeben werden, was wenig Aussagekraft hat.
	MacAlgorithms	Bei HMAC Schlüsseln der Algorithmus. Frage(n): 2
	ExpirationModel	Angabe wann der Schlüssel ungültig wird, aber nur bei externen Keys. Frage(n): -
	CloudHsmClusterId	Kann Aufschluss geben, ob der Schlüssel durch ein Cloud HSM verwaltet wird (Hardware-Sicherheitsmodul) Frage(n): 5

Tabelle 15: API Schlüsselverwaltung mit AWS KMS

Literaturverzeichnis

[1] Matthew Chiodi. *Cloud Threat Report - Putting the Sec in DevOps*. 2020. URL: https://www.paloaltonetworks.com/content/dam/pan/en_US/assets/pdf/reports/Unit_42/cloud-threat-report-spring-2020.pdf?utm_source=marketo&utm_medium=email&utm_campaign=Global-DA-EN-20-02-04-7010g000001JFyPAAW-P3-Prisma-unit-42-cloud-threat-report-%5BFY20%5D (besucht am 26. 02. 2023).

[2] NSA. *Mitigating Cloud Vulnerabilities*. Jan. 2020. URL: https://www.cisa.gov/uscert/ncas/current-activity/2020/01/24/nsa-releases-guidance-mitigating-cloud-vulnerabilities (besucht am 26. 02. 2023).

[3] IBM Corporation. *Cost of a Data Breach Report 2020*. Juli 2020. URL: https://www.ibm.com/security/digital-assets/cost-data-breach-report/1Cost%20of%20a%20Data%20Breach%20Report%202020.pdf (besucht am 26. 02. 2023).

[4] Hillary Baron u. a. *State of Cloud Security Risk, Compliance, and Misconfigurations*. Techn. Ber. Cloud Security Alliance, Sep. 2021. URL: https://cloudsecurityalliance.org/artifacts/state-of-cloud-security-risk-compliance/ (besucht am 26. 02. 2023).

[5] Peter M. Mell und Timothy Grace. *The NIST definition of cloud computing*. ed. Techn. Ber. 2011. DOI: 10.6028/nist.sp.800-145. URL: https://www.nist.gov/publications/nist-definition-cloud-computing.

[6] Bundesamt für Sicherheit in der Informationstechnik (BSI). *Cloud Computing Grundlagen*. de. URL: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Informationen-und-Empfehlungen/Empfehlungen-nach-Angriffszielen/Cloud-Computing/Grundlagen/grundlagen.html?nn=128880 (besucht am 26. 02. 2023).

[7] Neil MacDonald. *Innovation Insight for Cloud Security Posture Management*. Forschungsber. Gartner, Jan. 2019. URL: https://www.gartner.com/en/documents/3899373.

[8] David Puzas. *What is Cloud Security Posture Management (CSPM)?* en. Crowdstrike Inc., 24. Nov. 2021. URL: https://www.crowdstrike.com/cybersecurity-101/cloud-security/cloud-security-posture-management-cspm/ (besucht am 26. 02. 2023).

[9] Jyoti Bolannavar. „CSPM- Cloud Security Posture Management (Comprehensive Security for Cloud Environment)“. In: *International Journal of Scientific Research in Computer Science, Engineering and Information Technology* (Apr. 2020), S. 251–257. DOI: 10.32628/CSEIT206268. URL: https://www.researchgate.net/publication/346799564_CSPM-_Cloud_Security_Posture_Management_Comprehensive_Security_for_Cloud_Environment.

[10] AWS. *Modell der geteilten Verantwortung – Amazon Web Services (AWS)*. de-DE. 6. Feb. 2023. URL: https://aws.amazon.com/de/compliance/shared-responsibility-model/ (besucht am 26. 02. 2023).

[11] Frank Simorjay und Eric Tierling. „Shared Responsibility for Cloud Computing“. en. Version 2.0. In: *Microsoft Inc.* (Okt. 2019). URL: https://azure.microsoft.com/mediahandler/files/resourcefiles/shared-responsibility-for-cloud-computing/Shared%20Responsibility%20for%20Cloud%20Computing-2019-10-25.pdf (besucht am 26. 02. 2023).

[12] International Organization for Standardization (ISO). *ISO/IEC 27001 Standard – Information Security Management Systems*. en. Version 27001:2022. Okt. 2022. URL: https://www.iso.org/standard/27001 (besucht am 24. 02. 2023).

[13] Bundesamt für Sicherheit in der Informationstechnik (BSI). *BSI-Standard 200-2 - IT-Grundschutz Methodik*. de. Version 1.0. Okt. 2017, S. 180. URL: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/IT-Grundschutz/BSI-Standards/BSI-Standard-200-2-IT-Grundschutz-Methodik/bsi-standard-200-2-it-grundschutz-methodik_node.html (besucht am 26. 02. 2023).

[14] Bundesamts für Sicherheit in der Informationstechnik (BSI). *IT-Grundschutz-Kompendium (Edition 2022)*. Techn. Ber. Feb. 2022. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Kompendium/IT_Grundschutz_Kompendium_Edition2022.pdf?__blob=publicationFile&v=3#download=1.

[15] National Institute of Standards and Technology. *NIST Cybersecurity Framework Version 1.1 to NIST Special Publication 800-53, Revision 5*. Dez. 2020. URL: https://csrc.nist.gov/CSRC/media/Publications/sp/800-53/rev-5/final/documents/csf-pf-to-sp800-53r5-mappings.xlsx.

[16] Victoria Pillitteri u. a. *Security and Privacy Controls for Information Systems and Organizations (SP800-53)*. Techn. Ber. Sep. 2020. DOI: 10.6028/nist.sp.800-53r5. URL: https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final.

[17] The Center for Internet Security. *CIS Critical Security Controls (v8)*. en. Techn. Ber. Version 8. The Center for Internet Security, Mai 2021. URL: https://www.cisecurity.org/controls (besucht am 26. 02. 2023).

[18] NIST. *Framework for Improving Critical Infrastructure Cybersecurity (Cybersecurity Framework)*. en. Techn. Ber. Version 1.1. Cybersecurity Framework, 16. Apr. 2018. URL: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf (besucht am 26. 02. 2023).

[19] NIST. „The Five Functions“. en. In: *NIST* (Apr. 2018). URL: https://www.nist.gov/cyberframework/online-learning/five-functions (besucht am 21. 03. 2023).

[20] BSI. *Zuordnungstabelle ISO 27001 sowie ISO 27002 und IT-Grundschutz*. de. Version 4. Edition 2021. 3. Dez. 2021. URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Kompendium/Zuordnung_ISO_und_IT_Grundschutz.pdf?__blob=publicationFile&v=5 (besucht am 26. 02. 2023).

[21] Thomas Sager und Will Spier. *CIS Controls v8 Mapping to NIST SP 800-53 R5*. en. Version 8. Center for Internet Security, 10. Feb. 2022. URL: https://www.cisecurity.org/white-papers/cis-controls-v8-mapping-to-nist-800-53-rev-5/ (besucht am 26. 02. 2023).

[22] Thomas Sager, Tony Krzyzewski und Jackie Krzyzewski. *CIS Controls v8 Mapping to NIST CSF*. en. Version 8. Center for Internet Security, 11. Juni 2021. URL: https://www.cisecurity.org/white-papers/cis-controls-v8-mapping-to-nist-csf/ (besucht am 26. 02. 2023).

[23] Thomas Sager. *CIS Controls v8 Mapping to ISO/IEC 27002:2002*. en. Version 8. Center for Internet Security, 14. Dez. 2022. URL: https://www.cisecurity.org/white-papers/cis-controls-v8-mapping-to-iso-iec2-27002-2022 (besucht am 26. 02. 2023). draft.

[24] Amazon Web Services. *Amazon Elastic Compute Cloud API Reference*. en. Version 2016-11-15. URL: https://docs.aws.amazon.com/de_de/AWSEC2/latest/APIReference/ec2-api.pdf (besucht am 26. 02. 2023).

[25] Europäische Kommission. *Datenschutz-Grundverordnung (DSGVO) - Richtlinie (EU) 2016/680*. 27. Apr. 2016. URL: https://eur-lex.europa.eu/legal-content/DE/TXT/?uri=CELEX:32016L0680 (besucht am 26. 02. 2023).

[26] 13. Nationale Volkskongress der Volksrepublik China. *Cybersecurity Law (CSL) of the People's Republic of China (CSL)*. 1. Juni 2017. URL: http://www.cac.gov.cn/2016-11/07/c_1119867116.htm.

[27] 13. Nationale Volkskongress der Volksrepublik China. *Data Security Law of the People's Republic of China (DSL)*. 10. Juni 2021. URL: http://www.npc.gov.cn/englishnpc/c23934/202112/1abd8829788946ecab270e469b13c39c.shtml.

[28] California State Legislature. *California Consumer Privacy Act (CCPA)*. 28. Juni 2018. URL: https://leginfo.legislature.ca.gov/faces/codes_displayText.xhtml?lawCode=CIV&division=3.&title=1.81.5.&part=4.&chapter=&article.

[29] U.S. Department of Justice. *Clarifying Lawful Overseas Use of Data (CLOUD) Act*. März 2018. URL: https://www.justice.gov/dag/cloudact.

[30] Amazon Web Services. *Verifizierter Anbieter für Amazon AMIs*. de. 25. Sep. 2022. URL: https://docs.aws.amazon.com/de_de/AWSEC2/latest/UserGuide/sharing-amis.html (besucht am 26. 02. 2023).

[31] MITRE. *MITRE ATT&CK Matrix for Cloud IaaS*. en. Version v12. The MITRE Corporation. 25. Okt. 2022. URL: https://attack.mitre.org/matrices/enterprise/cloud/iaas/ (besucht am 26. 02. 2023).

[32] Gal Singer. *Threat Alert: Kinsing Malware Attacks Targeting Container Environments*. en-us. 4. März 2020. URL: https://blog.aquasec.com/threat-alert-kinsing-malware-container-vulnerability (besucht am 26. 02. 2023).

[33] BSI. *Technischen Richtlinie: Kryptographische Verfahren: Empfehlungen und Schlüssellängen*. de. Version 2022-01. Bundesamt für Sicherheit in der Informationstechnik, 28. Jan. 2022. URL: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/Technische-Richtlinien/TR-nach-Thema-sortiert/tr02102/tr02102_node.html (besucht am 26. 02. 2023).

[34] BSI. *Technischen Richtlinie: Kryptographische Verfahren: Empfehlungen und Schlüssellängen (Teil 4 SSH)*. de. Version 2022-01. Bundesamt für Sicherheit in der Informationstechnik, 24. Jan. 2022. URL: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/Technische-Richtlinien/TR-nach-Thema-sortiert/tr02102/tr02102_node.html (besucht am 26. 02. 2023).

[35] Erlend Oftedal. *Server Side Request Forgery*. en. Open Web Application Security Project. URL: https://owasp.org/www-community/attacks/Server_Side_Request_Forgery# (besucht am 26. 02. 2023).

[36] Colm MacCarthaigh. *Add defense in depth against open firewalls, reverse proxies, and SSRF vulnerabilities with enhancements to the EC2 Instance Metadata Service*. en. Amazon Web Services. 19. Nov. 2019. URL: https://aws.amazon.com/de/blogs/security/defense-in-depth-open-firewalls-reverse-proxies-ssrf-vulnerabilities-ec2-instance-metadata-service/ (besucht am 26. 02. 2023).

[37] Nick Frichette. *Steal EC2 Metadata Credentials via SSRF - Hacking The Cloud*. en. 1. Aug. 2020. URL: https://hackingthe.cloud/aws/exploitation/ec2-metadata-ssrf/ (besucht am 26. 02. 2023).

[38] Amazon Web Services. *AWS Well-Architected - Security Pillar*. 20. Okt. 2022. URL: https://docs.aws.amazon.com/pdfs/wellarchitected/latest/security-pillar/wellarchitected-security-pillar.pdf#welcome (besucht am 26. 02. 2023).

[39] Avik Mukherjee u. a. *AWS Security Reference Architecture*. Nov. 2022. URL: https://docs.aws.amazon.com/prescriptive-guidance/latest/security-reference-architecture/welcome.html (besucht am 26. 02. 2023).

[40] Amazon Web Services. *AWS Security Documentation*. en. 2022. URL: https://docs.aws.amazon.com/security/ (besucht am 26. 02. 2023).

[41] Amazon Web Services. *Bewährte Methoden für Sicherheit, Identität und Compliance (best-practises)*. 2022. URL: https://aws.amazon.com/de/architecture/security-identity-compliance/?cards-all.sort-by=item.additionalFields.sortDate&cards-all.sort-order=desc&awsf.content-type=*all&awsf.methodology=*all (besucht am 26. 02. 2023).

[42] Mike Wicks und Community. *CIS Amazon Web Services Foundations Benchmark*. en. Techn. Ber. Version 1.5.0. Center for Internet Security, 12. Aug. 2022. URL: https://www.cisecurity.org/benchmark/amazon_web_services (besucht am 26. 02. 2023).

[43] Brandon Evans, Eric Johnson und Wes Braga. *Secure Service Configuration in AWS, Azure and GCP*. Techn. Ber. 2021. SANS, Juni 2021. URL: https://sansorg.egnyte.com/dl/5STcKpQd8e (besucht am 26. 02. 2023).

[44] Aqua Security. *AWS Misconfiguration Database*. en-us. Aqua Security Software Ltd., Nov. 2022. URL: https://avd.aquasec.com/misconfig/aws/ (besucht am 26. 02. 2023).

[45] Bridecrew Inc. *AWS Policy Index*. en. 2021. URL: https://docs.bridgecrew.io/docs/s3_16-enable-versioning (besucht am 26. 02. 2023).

[46] Bridecrew Inc. *GitHub Repository Bridgecrew - AWS Checks*. 7. Nov. 2022. URL: https://github.com/bridgecrewio/checkov/tree/master/checkov/terraform/checks/resource/aws (besucht am 26. 02. 2023).

[47] Erez Shinan. *Lark - a parsing toolkit for Python*. en. Version 45. Mai 2023. URL: https://github.com/lark-parser/lark (besucht am 18. 05. 2023).

[48] *Backus-Naur-Form*. de. Page Version ID: 228359573. Nov. 2022. URL: https://de.wikipedia.org/w/index.php?title=Backus-Naur-Form&oldid=228359573 (besucht am 18. 05. 2023).

[49] Styra. *Open Policy Agent - Rego Language Reference (v0.51.0)*. en. Cloud Native Computing Foundation, 31. März 2023. URL: https://www.openpolicyagent.org/docs/latest/policy-reference/ (besucht am 02. 04. 2023).

[50] Torin Sandall. *The Rego Playground*. en. Version 0.52.0. Styra Inc. 8. März 2023. URL: https://play.openpolicyagent.org/ (besucht am 06. 05. 2023).

[51] G. Carpenter, Randy Mowen und Robin Regnier. *CIS Controls v8 Cloud Companion Guide*. Techn. Ber. The Center for Internet Security, März 2022. URL: https://www.cisecurity.org/insights/white-papers/cis-controls-v8-cloud-companion-guide.

[52] Amazon Web Services. *AWS-Dokumentation*. 28. Sep. 2022. URL: https://docs.aws.amazon.com/en_us/index.html.

[53] Amazon Web Services. *Getting Started with Amazon Web Services in China*. 28. Sep. 2022. URL: https://docs.amazonaws.cn/en_us/aws/latest/userguide/introduction.html.

Bildverzeichnis

Tabellenverzeichnis

Listingverzeichnis

Abkürzungsverzeichnis

ACL	Access Control List. 26, 50–52, 121, 171
AMI	Amazon Machine Image. 27, 31, 32, 34, 35, 62, 100, 112, 167, 168
API	Application Programming Interface. iii, iv, 3, 5, 16, 19–21, 24–27, 29, 31, 38, 41–44, 47–50, 53, 55–60, 62, 63, 68, 74, 88, 90, 100, 112–114
ARN	Amazon Resource Name. 168
AWS	Amazon Web Services. 11, 24, 26–29, 31, 32, 39, 40, 58, 61, 75, 79, 80, 85, 112
AZ	Availability Zone. 53
Azure	Microsoft Azure. 24, 39, 61
BSI	Bundesamt für Sicherheit in der Informationstechnik. 12, 13, 17, 39, 101, 107, 110
CI/CD	Continuous Integration/Continuous Delivery. 94
CIS	Center for Internet Security. 12, 14, 74–76, 80, 81, 101, 102, 109, 110
CLI	Command Line Interface. 76
CNCF	Cloud Native Computing Foundation. 93
CSA	Cloud Security Alliance. 2
CSF	Cybersecurity Framework. 14, 101–103, 105, 111
CSP	Cloud Service Provider. iv, 6–8, 11, 15, 16, 19, 20, 26, 36, 40, 58–60, 72, 79, 90, 108, 127
CSPM	Cloud Security Posture Management. iii, iv, 3–5, 8–10, 12, 14–22, 25, 26, 47, 55, 56, 59, 60, 64, 68, 69, 73, 74, 77–79, 83, 85, 87, 89–93, 95, 101, 102, 106–111, 113, 114, 121, 124, 127, 128, 181
CVE	Common Vulnerabilities and Exposures. 78
EBS	Elastic Block Store. 43, 45, 66, 68, 70
EC2	Elastic Cloud Computing. 27, 31
EFS	Elastic File System. 44–48, 55, 66, 68, 71, 85, 86
FaaS	Function as a Service. 7, 8, 26, 121, 181
GCP	Google Cloud Platform. 24, 39, 61

HCL HashiCorp Configuration Language. 60, 61, 87

IaaS Infrastructure as a Service. 3–7, 9, 10, 12, 25, 26, 32, 42, 64, 69, 73, 74, 79, 80, 83, 109, 120, 121, 181
IaC Infrastructure as Code. iii, iv, 20, 59, 60, 79, 85–87, 89, 93, 113, 127, 193
IAM Identity und Access Management. 26, 41, 47, 56, 72, 74, 87, 100, 121, 168, 171
IGW Internet Gateway. 55
IMDS Instanz-Metadaten-Service. 39, 40, 168
ISMS Information Security Management System. 13, 17

JSON JavaScript Object Notation. 61

KMS Key Management Service. 58

MFA Multifaktor-Authentifizierung. 66, 76

NACL Network Access Control List. 54
NFS Network File System. 44
NIST National Institute of Standards and Technology. 8, 12, 13, 101–103, 105, 106, 110
NSA National Security Agency. 2

OPA Open-Policy-Agent. 93, 94, 98–100, 113, 138, 143, 193
OWASP Open Web Application Security Project. 40

PaaS Platform as a Service. 6, 7, 9, 12, 76, 79, 80, 99, 114

RDS Relational Database Service. 66

S3 Simple Storage Service. 49, 66, 72, 80, 86
SaaS Software as a Service. 7, 9, 10, 12, 79, 80, 99, 114
SDK Software Development Kit. 24, 27, 72, 87
SG Security Group. 55, 85, 87
SRA AWS Security Reference Architecture. 65, 68, 69
SSH Secure Shell. 36
SSRF Server Side Request Forgery. 40

TF	Terraform. 60, 63
TLS	Transport Layer Security. 45
VM	Virtual Machine. 33, 36, 39, 40
VPC	Virtual Private Cloud. 45, 53–55, 69, 85
WAF	Web Application Firewall. 55, 56
Web-ACL	web access control list. 55, 56

Glossar

Amazon Machine Image	Ein Amazon Machine Image (AMI) ist ein unterstütztes und verwaltetes Abbild (Image) für virtuelle Maschinen, das von AWS bereitgestellt wird und die Informationen liefert, die zum Starten einer Instanz einer virtuellen Maschine erforderlich sind. Beim Starten einer Instanz muß ein AMI angeben werden. Es können eigene, angepaßte AMIs erstellt werden. (Quelle: AWS). 25, 66, 120
Amazon S3 Glacier	Amazon S3 Glacier (S3 Glacier) ist ein dauerhafter und sklalierbarer Speicherdienst für eine ökonomische Datenarchivierung oder Langzeitbackups. Es handelt sich um einen Cloud-Objektspeicher. Daten können mit S3 Glacier für Monate, Jahre oder sogar Jahrzehnte gespeichert werden. Preislich ist der Speicherdienst dabei im Vergleich zu anderen AWS Speicherdiensten, insbesondere der Storage-Preisanteil, günstiger. Daten in Transfer von oder zu S3 Glacier können verschlüsselt werden, sowie wird die Datenverschlüsselung in Ruhe (at rest) unterstützt. Derzeit erfolgt dies serverseitig mit 256-Bit Advanced Encryption Standard (AES-256). Auch die in S3 angebotene clientseitige Verschlüsselung vor dem Speichern von Daten in S3 Glacier wird unterstützt.(Quelle: AWS). 120
AWS Account	Ein AWS-Konto (Account) ist ein Container für AWS-Ressourcen. Ein gängiger Cloud-Begriff hierfür ist der Tenant, als eine isolierte und abgeschlossene Mieteinheit in einer Public-Cloud. Das AWS-Konto stellt auch die Verwaltungsfunktionen für den Zugriff und die Fakturierung bereit. Der Account ist nicht mit dem AWS-Benutzer zu verwechseln. Ein Unternehmen kann mehrere Accounts bei AWS verwalten, aus unterschiedlichen Gründen - z.B. zur Isolation der Ressourcen aus Sicherheitsgründen, zur Flexibilisierung der Teams, oder auch aus Gründen der Finanzbuchhaltung. (Quelle: AWS). 25, 30, 120

AWS Backup	AWS Backup ist ein verwalteter und zentralisierter Dienst zur Automatisierung von Datensicherungen für mehrere AWS Dienste. AWS Backup unterstützt derzeit die AWS Dienste EC2, EBS, RDS, EFS, DynamoDB, FSx und Storage Gateway. AWS Backup ermöglicht das erstellen und Datensicherungsplänen, zeitlichen Abläufen und Datenaufbewahrungsregeln (retention rules). Die Überwachung (Monitoring), sowie die Erstellung von Audit- und Compliance-Berichten wird angeboten. Der Vorteil bei der Nutzung von AWS Backup ist die zentrale Verwaltung der Datensicherung und dies auch über mehrere AWS Dienste im Verbund. (Quelle: AWS). 120
AWS Outpost	AWS Outposts ist ein vollständig verwalteter Service, der AWS-Infrastruktur, Services, APIs und Tools auf Kundenstandorte ausweitet. Ein Outpost ist ein Pool von AWS-Rechen- und Speicherkapazitäten, die an einem Kundenstandort bereitgestellt werden. AWS betreibt, überwacht und verwaltet diese Kapazität als Teil einer AWS-Region. (Quelle: AWS). 49, 120
EC2	Amazon Elastic Compute Cloud (Amazon EC2) bietet skalierbare Rechenkapazität in der Amazon Web Services Cloud (AWS Cloud). Mit Amazon EC2 können so viele oder so wenige virtuelle Server wie benötigt gestartet werden, die Sicherheit und das Netzwerk konfiguriert und der Speicher verwaltet werden. Amazon EC2 ermöglicht, hoch- und runterzuskalieren, um Anforderungsänderungen oder Nutzungsspitzen zu bewältigen. (Quelle: AWS). 30, 118
Elastic Block Store	Amazon Elastic Block Store (Amazon EBS) bietet virtualisierte Datenträger (Volumes) für die Speicherung auf Blockebene, die in Verbindung mit EC2-Instanzen gemountet und verwendet werden. EBS-Volumes verhalten sich wie unformatierte Blockgeräte. EBS-Volumes werden in einer ausgewählten Availability Zone bereitgestellt. (Quelle: AWS). 25, 43, 70, 120

Elastic File System	Amazon Elastic File System (EFS) bietet ein elastisches (skalierbares) Dateisystem zur Verwendung mit AWS Cloud-Dienstleistungen und lokalen Ressourcen. Es wächst und schrumpft automatisch beim Hinzufügen und Entfernen von Dateien. Amazon EFS unterstützt das Network File System (NFSv4) Protokoll. Mehrere Compute-Instanzen (EC2, ECS und Lambda) können gleichzeitig zugreifen. (Quelle: AWS). 25, 43, 44, 120
Instance Metadata Service	Instance-Metadaten sind Daten über eine Instance. Diese Metadaten können genutzt werden die EC2 Instanz zu konfigurieren. Instance-Metadaten werden typischerweise in sogenannte Kategorien unterteilt. Die Metadaten einer Instanz können aus der laufenden virtuellen Umgebung über einen nicht routingfähige Adressierung abgerufen werden (`http://169.254.169.254/latest/meta-data/`.) (Quelle: AWS). 40, 41
Internet-Gateway	Ein Internet-Gateway (IGW) ist eine horizontal skalierte, redundante und hochverfügbare VPC-Komponente, die die Kommunikation zwischen einer Virtual Private Cloud (VPC) und dem Internet ermöglicht. Es unterstützt IPv4- und IPv6-Datenverkehr. Es verursacht keine Verfügbarkeitsrisiken oder Bandbreitenbeschränkungen für den Netzwerkverkehr. Über ein Internet-Gateway können Ressourcen in öffentlichen Subnetzen (wie EC2-Instances) eine Internetverbindung herstellen, sofern sie über eine öffentliche IPv4-Adresse oder eine IPv6-Adresse verfügen (egress / outbound). Desgleichen können Ressourcen im Internet über die öffentliche IPv4-Adresse oder die IPv6-Adresse eine Verbindung mit Ressourcen im Subnetz initiieren (inbound). (Quelle: AWS). 53, 191

Simple Storage Service	Amazon Simple Storage Service (Amazon S3) ist ein Objektspeicher-Service. Objekte werden dabei in sogenannten *Buckets* verwaltet, die durch den Cloud-Nutzer erzeugt werden. Die Benennung der Buckets ist dabei relativ frei, muss jedoch innerhalb der globalen Amazon S3 Infrastruktur eindeutig sein. Übergreifend über alle Kunden. Ein Bucket wird in einer Region erstellt, was Auswirkungen auf die Latenzzeit hat, als auch bezüglich behördliche und rechtlichen Vorschriften zu beachten ist. Amazon S3 bietet eine Versionierung der Daten an, was einen positiven Effekt auf die Verfügbarkeit und Sicherheit der Daten haben kann. Zugriffsrechte auf die Buckets und Objekte werden über Zugriffssteuerungslisten (Access-Control-List (ACL)) gesteuert. Eine serverseitige Verschlüsselung, zur Verschlüsselung der S3 Objekte, ist möglich. Ebenso ist eine clientseitige Verschlüsselung möglich, bei der die Daten bevor diese zu S3 hochgeladen werden, auf dem Client verschlüsselt werden. AWS bietet hierzu das sogenannte AWS Encryption SDK an. (Quelle: AWS). 25, 120
Transit Gateway	AWS Transit Gateway fungiert als Cloud-Router zur Vereinfachung der Netzwerkarchitektur. AWS Transit Gateway verbindet Amazon Virtual Private Clouds (VPCs), als auch On-Premises-Netzwerke über einen zentralen Hub. Damit wird das Netzwerkmanagement vereinfacht und Direktverbindungen (Peer-to-Peer) ersetzt. Die Daten werden dabei automatisch verschlüsselt und werden nie über das öffentliche Internet übertragen. (Quelle: AWS). 121

Virtual Private Cloud	Mit Amazon Virtual Private Cloud (Amazon VPC) können AWS-Ressourcen in einem virtuellen Netzwerk betrieben werden. Dieses virtuelle Netzwerk entspricht weitgehend einem herkömmlichen Netzwerk, wie es in einem Rechenzentrum betrieben wird, kann jedoch die Vorzüge der skalierbaren Infrastruktur von AWS nutzen. In einer VPC können IPv4- und IPv6-Adressen genutzt werden. Neben privaten IP-Adressen, können auch öffentliche IPv4 Adressen den Ressourcen zugewiesen werden. Ein VPC kann in Subnetze untergliedert werden, die Teilbereiche des VPC-Netzes abbilden. Routing-Tabellen bestimmen die Wegeleitung des Netzwerkverkehrs. Ein Gateway verbindet die VPC it anderen Netzwerken (z.B. Internet-Gateway zur Konnektivität in das Internet) (Quelle: AWS). 167

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe.
Die Stellen der Arbeit, die anderen Quellen im Wortlaut oder dem Sinn nach entnommen wurden, sind durch Angaben der Herkunft kenntlich gemacht. Dies gilt auch für Zeichnungen, Skizzen, bildliche Darstellungen sowie für Quellen aus dem Internet.

Ich erkläre ferner, dass ich die vorliegende Arbeit in keinem anderen Prüfungsverfahren als Prüfungsarbeit eingereicht habe oder einreichen werde.

Die eingereichte schriftliche Fassung entspricht der auf dem Medium gespeicherten Fassung.

Tuttlingen, 26. Juni 2023
Ort, Datum *Unterschrift*

Thesen

Master-Thesis
Möglichkeiten und Grenzen der automatisierten Detektion und Mitigation von Schwachstellen in Cloud-Konfigurationen mittels Cloud Security Posture Management

Eingereicht am:	26. Juni 2023
von:	Andreas Gollwitzer
Betreuer:	Prof. Dr. Nils Gruschka
Zweitbetreuerin:	Prof. Dr. Antje Raab-Düsterhöft

- Die Sicherheitsanalyse der Infrastructure as Code (IaC) Daten kann einen erheblichen Sicherheitsvorteil bieten, im Gegensatz zur Cloud-API basierten Analyse. Mängel können vor einer Installation in der Cloud-Umgebung identifiziert werden.
- Der Aufwand für die Erstellung von automatisierten Sicherheitsregeln ist als nicht gering, teils sogar hoch, zu werten und bedarf spezieller Expertise.
- Die Wiederverwendbarkeit automatisierter Sicherheitsregeln könnte durch den Einsatz von Standard-Policy Werkzeugen, wie dem Open-Policy-Agent (OPA), gefördert werden. Für einen Austausch - Wissen und Sourcecode - zwischen Unternehmen kann dies hilfreich sein.
- Eine automatisierte Behebung von Sicherheitsmängeln ist oftmals nur sehr eingeschränkt möglich, da komplexere Vorgänge notwendig sind. Die frühe Vermeidung ist durch einen Ansatz vor Installation umso relevanter.
- Die Kombination aus einer automatisierten Sicherheitsauditierung vor und nach Installation verspricht den größten Mehrwert, da Konfigurationsänderungen nach der Installation zusätzlich erkannt werden könnten.