Technology
Arts Sciences
TH Köln

5G-FORAN
DFIR IN OPEN RAN

Implementierung eines Dashboards für Pentesting in einer Digital Forensics und Incident Response (DFIR) Umgebung für Open RAN

Bachelorarbeit zur Erlangung des akademischen Grades
Bachelor of Science
im Studiengang Technische Informatik
an der Fakultät für Informations-, Medien- und Elektrotechnik
der Technischen Hochschule Köln

vorgelegt von: Jurek Miklas Jesse
Matrikel-Nr.: 11135480

Erstgutachter: Prof. Dr. Andreas Grebe
Zweitgutachter: Henrik Wittemeier, B.Sc.

Wiesbaden, 17.12.2024

Abstract

The dashboard developed in this work was specifically designed for use in Digital Forensics und Incident Response (DFIR) environments for Open Radio Access Network (Open RAN), supporting penetration testing scenarios and enabling precise vulnerability assessments. The integration of an empirical method and the use of CVSS data facilitate detailed analyses and visualizations. The dashboard provides a tool for visualizing and analyzing vulnerabilities in the highly virtualized Open RAN environment. Empirical evidence demonstrates which attack techniques pose significant risks or offer higher potential from an attacker's perspective. The same findings were achieved in relation to O-RAN, where empirical and statistical methods identified components vulnerable to specific techniques. This work makes a significant contribution to security research while offering practical approaches for future developments and investigations in DFIR contexts.

Keywords: Open RAN, O-RAN, DFIR, 5G, Dashboard

Kurzfassung

Das in dieser Arbeit entwickelte Dashboard wurde speziell für den Einsatz in Digital Forensics und Incident Response (DFIR)-Umgebungen für Open Radio Access Network (Open RAN) entworfen, um Pentesting-Szenarien zu unterstützen und Schwachstellen präzise zu bewerten. Die Integration einer empirischen Methode und die Nutzung von CVSS-Daten ermöglichen detaillierte Analysen und Visualisierungen. Das Dashboard bietet ein Werkzeug zur Visualisierung und Analyse von Schwachstellen im stark virtualisierten Open RAN Umfeld. Es wird empirisch aufgezeigt, welche Angriffstechniken besonders risikobehaftet sind, beziehungsweise aus der Perspektive der Angreifenden ein höheres Potenzial bieten. Dasselbe Ergebnis wurde auch in Bezug auf O-RAN erreicht. Über empirische und statistische Methoden werden O-RAN Komponenten identifiziert, die bei Ausnutzung von spezifischen Techniken gefährdet sind. Das Ergebnis dieser Arbeit stellt einen wichtigen Beitrag zur Sicherheitsforschung dar und bietet zugleich Ansätze für zukünftige Entwicklungen und Untersuchungen in DFIR-Kontexten.

Schlüsselwörter: Open RAN, O-RAN, DFIR, 5G, Dashboard

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Listingverzeichnis

Abkürzungsverzeichnis

5G-FORAN IT-Forensik und Behandlung von IT-Sicherheitsvorfällen im Open RAN. 1, 2, 7, 15, 47

ACEMA A Comprehensive Empirical Method to Analyze Threats in O-RAN Environments. 2, 6, 13, 15–17, 20–23, 25–28, 32, 39, 42–44, 46, 47, 74

AT ATTACK-Tool. 2, 9, 13, 15–17, 22, 25, 32, 37, 43, 46

ATLAS Adversarial Threat Landscape for Artificial-Intelligence Systems. 10

ATT&CK Adversarial Tactics, Techniques, and Common Knowledge. 1, 2, 6, 10, 12, 13, 15, 25, 35, 44, 47, 68

CAPEC Common Attack Pattern Enumeration and Classification. 6, 11, 12, 16, 23, 25, 26, 32, 44, 45

CLI Command Line Interface. 9, 13, 20, 23, 32

CSS Cascading Style Sheets. 9

CTI Cyber Threat Intelligence. 13, 16, 23, 44

CVE Common Vulnerabilities and Exposures. 3, 6, 10–17, 25–29, 31, 32, 39, 40, 42–44

CVSS Common Vulnerability Scoring System. 2, 6, 11, 14–16, 21, 26–30, 39–41, 43–45, 47, 74

CWE Common Weakness Enumeration. 6, 11–13, 16, 25, 26, 31, 32

DFIR Digital Forensics und Incident Response. 1, 2, 15, 41, 47, 48, I, II

DOT Directed acyclic graph of tommorow. 33, 34

ENISA The European Union Agency for Cybersecurity. 5

HTML Hypertext Markup Language. 9, 30, 31, 33, 34, 36, VII

JS JavaScript. 9, 33, 34, 36

KTM Kubernetes Threat Matrix. 13, 43, 44

MITRE MITRE Corporation. 1–3, 6, 7, 10, 12, 13, 15–17, 22, 23, 25–28, 32, 35, 36, 40–42, 44–47, 68

NIS Network and Information Systems Cooperation Group. 5

NIST National Institute of Standards and Technology. 13, 16, 43

NVD National Vulnerability Database. 13, 14, 16, 25, 43, 44

O-DU O-RAN Radio Unit. 41

O-RU O-RAN Central Unit. 41

Open RAN Open Radio Access Network. 1–3, 5–7, 9, 12, 40, 46–48, I, II

RAN Radio Access Network. 1, 7

RIC RAN Intelligent Controller. 41

SC Software Community. 2, 3, 7

STIX Structured Threat Information Expression. 23

TM4K Threat Matrix for Kubernetes. 12, 13, 15, 16, 27, 35, 40, 43, 44

VM virtuelle Maschine. 20, 47

WG1 Use Cases and Overall Architecture Group. 7

WG11 Security Work Group. 3, 4, 13, 17, 35, 40, 41

WORM Write-Once, Read-Many. 27

1. Einleitung

Die Einführung des 5G-Netzwerks stellt einen bedeutenden Schritt in der Telekommunikationslandschaft dar und eröffnet durch die erhöhte Flexibilität und Geschwindigkeit neue Anwendungsfelder. 5G ist ein Mobilfunkstandard der fünften Generation, der die Möglichkeiten bietet, eine deutlich schnellere Datenübertragung und extrem niedrige Latenzzeiten zu erreichen und dabei eine Vielzahl von Geräten gleichzeitig zu vernetzen, was sowohl Privatnutzer als auch Industrieanwendungen revolutioniert. Eine zentrale Entwicklung hierbei ist das Konzept von Open RAN, welches durch die offene Spezifikation und modulare Architektur das herkömmliche Radio Access Network (RAN)-Design revolutioniert [1]. Open RAN ermöglicht eine Interoperabilität zwischen verschiedenen Herstellern und eine flexiblere Verwaltung der Netzwerkkomponenten, was den Netzwerkbetreibern potenzielle Kosteneinsparungen und eine größere Anpassungsfähigkeit bietet. Die Abhängigkeit von proprietären Herstellern wie Huawei, Ericsson und Nokia fallen dabei weg [2]. Die O-RAN Alliance, ein Konsortium aus Telekommunikationsanbietern, Herstellern und Forschungsinstitutionen, treibt diesen Ansatz voran und entwickelt Standards und Architekturen, die speziell für 5G und zukünftig auch für 6G ausgelegt sind. Diese Standards sind auf die Entkopplung von Hardware und Software ausgelegt und fördern die Virtualisierung von RAN-Komponenten, um so die Flexibilität der Netzbetreiber zu erhöhen [3].

Das Forschungsprojekt IT-Forensik und Behandlung von IT-Sicherheitsvorfällen im Open RAN (5G-FORAN), eine Kooperation zwischen der Technischen Hochschule Köln und PROCYDE GmbH, widmet sich den sicherheitstechnischen Herausforderungen im O-RAN Umfeld, insbesondere im Kontext der DFIR [4]. Ziel des Projekts ist die Entwicklung von Methoden zur Analyse, Behandlung und Behebung von Sicherheitsvorfällen in Open RAN-Netzwerken. Hierbei wird der gesamte Lebenszyklus eines Sicherheitsvorfalls berücksichtigt: von der Erkennung über die Analyse bis hin zur forensischen Nachverfolgung. Das Projekt liefert wissenschaftliche Erkenntnisse zur Sicherheit von Open RAN-Systemen, indem neue Ansätze zur Analyse und Behandlung von Sicherheitsvorfällen entwickelt werden oder auf existierender Forschung aufgebaut wird. Die Verknüpfung empirischer Daten mit bewährten Sicherheitsmodellen wie dem MITRE Corporation (MITRE) Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK)-Framework schafft eine wissenschaftliche Basis zur Bewertung von Risiken. Die Forschung trägt dazu bei,

das Verständnis für die Bedrohungslandschaft in virtualisierten Mobilfunknetzen zu vertiefen und liefert Ansätze für die Weiterentwicklung von sicheren Implementierungen der O-RAN-Spezifikationen. Der wissenschaftliche Wert des Projekts liegt insbesondere in der Übertragbarkeit der entwickelten Methoden auf andere virtualisierte oder containerisierte Systeme sowie der Schaffung einer Grundlage für weiterführende Arbeiten im Bereich 5G- und 6G-Sicherheit.

Das Forschungsprojekt 5G-FORAN ist in zwei Teilprojekte unterteilt: 5G-FORAN-ATTACK und 5G-FORAN-DFIR. Im Rahmen von 5G-FORAN-ATTACK wird ein System entwickelt, das gezielte Angriffssimulationen auf Open RAN-Komponenten ermöglicht, um spezifische Angriffsspuren zu generieren. Diese Angriffsspuren können dann mit - im Teilprojekt 5G-FORAN-DFIR - entwickelten Methoden identifiziert und behandelt werden.

In dieser Arbeit liegt der Fokus auf der Weiterentwicklung und Weiterimplementierung eines Dashboards, welches als zentrales Tool für die Visualisierung und Analyse von Angriffssimulationen fungiert [4,5]. Das Dashboard stellt eine Komponente im ATTACK-Teilvorhaben von 5G-FORAN dar. Zu den spezifischen Anforderungen zählen die Darstellung von Angriffspfaden und die Integration des Common Vulnerability Scoring System (CVSS)s zur Bedrohungsbewertung. Außerdem wurde speziell für die Kategorisierung der Angriffe eine abgewandelte Version der MITRE ATT&CK-Matrix erstellt und die Daten empirisch eingeordnet. Die Arbeit der MITRE-Corporation spielt eine sehr wichtige Rolle bei der Kategorisierung von Angriffen und der Bewertung von Schwachstellen. MITRE stellt unter anderem mit dem ATT&CK-Framework eine umfassende strukturierte Sammlung von Angriffstechniken und -taktiken bereit [6,7]. Dieses Framework unterstützt eine analytische Herangehensweise an die Bewertung der Angriffe Open RAN [8]. Diese Arbeit implementiert die Methode, die Klement, Liu und Katzenbeisser in ihrem Paper *Toward Securing the 6G Transition: A Comprehensive Empirical Method to Analyze Threats in O-RAN Environments*[1] beschreiben, um Schwachstellen in Open RAN-Komponenten anhand empirischer Daten zu bewerten. Es wird hierbei keine Bewertung aller möglichen Angriffe vorgenommen, sondern die durch das Teilvorhaben 5G-FORAN-ATTACK gefundenen und durch das ATTACK-Tool (AT) simulierten Angriffe auf eine Referenzimplementierung von Open RAN der O-RAN Software Community (SC) bewertet. Die Bewertung nach dem CVSS wird in der Übersicht einzelner Angriffe und in der modifizierten MITRE-Matrix dargestellt [8,9].

[1] Von den Authoren wird ein Akronym verwendet: A Comprehensive Empirical Method to Analyze Threats in O-RAN Environments (ACEMA). Das Akronym wird in dieser Arbeit übernommen.

2. Stand der Wissenschaft

Seit den Anfängen der Entwicklung von Open RAN im Februar 2016 und der Gründung der O-RAN Alliance 2018 gibt es einige wissenschaftliche Arbeiten, die sich mit dem Thema Schwachstellenanalyse und Schwachstellenbewertung in einer Open RAN Umgebung beschäftigen [10]. Unter dem Stichwort „O-RAN“ finden sich über die MITRE Common Vulnerabilities and Exposures (CVE) Suche aktuell 12 Einträge, die Schwachstellen in der O-RAN-SC-Referenzimplementierung betreffen [11]. Die systematische Suche nach und Analyse von Schwachstellen ist entscheidend, um Sicherheitslücken frühzeitig zu erkennen und geeignete Gegenmaßnahmen zu implementieren. In diesem Kapitel wird ein Überblick über den Forschungsstand anhand einer Auswahl von Arbeiten gegeben.

2.1. Technischer Report der O-RAN Alliance Security Work Group

Die Security Work Group (WG11) der O-RAN Alliance beschäftigt sich mit den sicherheitstechnischen Aspekten von Open RAN und veröffentlicht in regelmäßigen Abständen einen technischen Report, der eine Bedrohungsmodellierung und Risikobeurteilung enthält. Zum Zeitpunkt der Veröffentlichung ist der Report in Version *v04.00* die neueste Fassung dieses Dokuments. Der Report umfasst sowohl mögliche O-RAN-spezifische Angriffe auf Komponenten und Schnittstellen in einem Open RAN-System, als auch nicht O-RAN-spezifische Bedrohungen, wie Supply-Chain-Angriffe auf quell-offenen Programmcode oder physischen Eingriff um Zugriff auf sensible Daten zu erlangen, die in allen Open RANs existieren können [12]. Im Folgenden wird ein Überblick über die O-RAN-spezifischen Bedrohungen gegeben. Die davon betroffenen Komponenten und Schnittstellen sind in Abbildung 3.1 dargestellt.

Es wird betrachtet wie hoch die Wahrscheinlich (*Likelihood*) ist, dass eine Schwachstelle (*Vulnerability*) durch eine spezifische Bedrohung (*Threat*) ausgenutzt wird, und welche Konsequenzen (*Consequences*) dies für das Asset hat. Die O-RAN Alliance definiert als Basis für die Risikobewertung eine Reihe von Einflussfaktoren, die in Abbildung 2.1 dargestellt sind. Ein Asset ist dabei eine (Teil-)Komponente

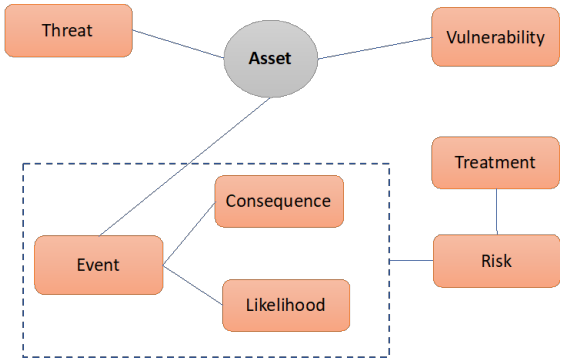

Abbildung 2.1.: Einflussfaktoren für die Risikobeurteilung (Quelle: [12])

oder eine Schnittstelle zwischen Komponenten. Die WG11 identifiziert dazu 35 Bedrohungen in der Komponente *O-RAN Cloud* und 69 Bedrohungen die auf alle anderen Komponenten und Schnittstellen zutreffen [12]. Die O-RAN Alliance definiert in ihrem Bericht 102 kritische Assets, also solche, die besonders vor Beeinflussung in den Bereichen Integrität, Verfügbarkeit, Vertraulichkeit, Wiederholbarkeit und Authentizität geschützt werden müssen. Die Verbindung zwischen Bedrohungen und kritischen Assets wird im Bedrohungsinventar hergestellt. Aus der Schweregradbeurteilung und der Eintrittwahrscheinlichkeitsbeurteilung wird der finale Risikowert berechnet. Die Verbindung zwischen einer spezifischen Bedrohung und dem zugehörigen Risikowert stellt das Ergebnis der Risikobewertung dar.

Dieser Report gibt keine subjektive Bewertung darüber ab, wie risikobehaftet die Komponenten im System sind. Es handelt sich um eine rein objektive Bewertung, die durch einen Risikowert festgelegt ist. Der Risikowert setzt sich zusammen aus dem Schweregrad und der Wahrscheinlich der Ausnutzung. Der Großteil der Bedrohungen wird dabei in der Risikobewertung mit einem Risikowert von *High* eingestuft, vergleiche Abbildung 2.2 [12].

2.2. BSI Risikoanalyse

In der Studie des Bundesamts für Sicherheit in der Informationstechnik (BSI) wird eine weitere Risikoanalyse durchgeführt. Die Studie zielt dabei nicht darauf ab, spezifische Schwachstellen in der Implementierung, sondern in den Spezifikationen der O-RAN Alliance zu finden. Köpsell et al. führen an, dass die veröffentlichten O-RAN-Spezifikationen zum Zeitpunkt Februar 2022 nicht viele Vorgaben zur Sicherheit machen und sich insbesondere nicht an dem Ansatz *security/privacy by design/default* orientiert [1]. Als Folge dessen werden viele potenzielle

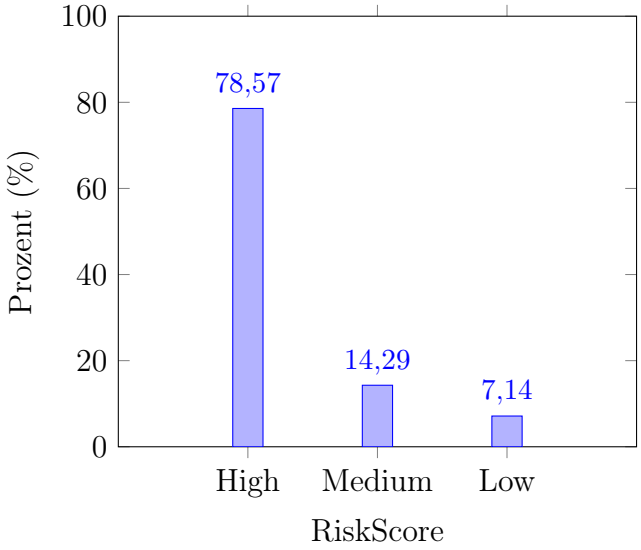

Abbildung 2.2.: Risikobewertung der Bedrohungen in O-RAN Komponenten und Schnittstellen

Sicherheitsrisiken mit mittlerem bis hohem Schweregrad festgestellt. Die Studie präsentiert Maßnahmen, deren Umsetzung zur Verbesserung der Sicherheit in der O-RAN-Umgebung führen würde. Köpsell et al. betonen dabei die Dringlichkeit, diese Maßnahmen in einem möglichst frühen Stadium in den Spezifikationen zu berücksichtigen [13].

Weitere umfangreiche Risikoanalysen für 5G oder 5G RAN wurden von Network and Information Systems Cooperation Group (NIS) und The European Union Agency for Cybersecurity (ENISA) veröffentlicht, diese sind aber nicht auf Open RAN fokussiert [14,15].

2.3. Empirische Analyse von Schwachstellen im Open RAN Umfeld

Klement, Liu und Katzenbeisser untersuchen in ihrem Artikel *Toward Securing the 6G Transition: A Comprehensive Empirical Method to Analyze Threats in O-RAN Environments* (ACEMA) die Sicherheitsherausforderungen, die mit dem Übergang von 5G zu 6G-Netzen und der Einführung von Open RAN-Technologien einhergehen. Ziel ihrer Forschung ist die Entwicklung eines umfassenden Ansatzes zur Analyse von Sicherheitsbedrohungen in Open RAN Umgebungen. Hierzu kombinieren die

Autoren das MITRE ATT&CK Framework mit empirischen Daten, um Bedrohungen in einer O-RAN-Implementierung zu analysieren. Im Zentrum ihrer Methodik steht die Abbildung einer MITRE-Technik zu einem spezifischen CVE-Datum über das Durchsuchen der unterschiedlichen Kategorisierungssysteme Common Attack Pattern Enumeration and Classification (CAPEC), Common Weakness Enumeration (CWE) und CVE. Die Anwendung des CVSSs, um den Schweregrad möglicher Schwachstellen zu bewerten, geht über die bisher vorgenommenen Bewertungssysteme der O-RAN Alliance heraus und ermöglicht eine granuläre Auswertung der Ergebnisse. Die Erkenntnisse von Klement, Liu und Katzenbeisser tragen dazu bei, ein erhöhtes Bewusstsein für spezifische Bedrohungsszenarien und vulnerable O-RAN Komponenten zu schaffen. Dies wird durch einfach verständliche Datenvisualisierungen erreicht [9].

2.4. Empirische Analyse von Schwachstellen im Android Umfeld

Auch in anderen Feldern der Informatik werden empirische Methoden genutzt, um die Sicherheit von Komponenten im jeweiligen System zu analysieren und bewerten.

Mit „The Android OS Stack and Its Vulnerabilities“ wurde 2019 eine empirische Studie durchgeführt die, ähnlich wie ACEMA im Umfeld von Open RAN, eine umfangreiche Schwachstellenanalyse und Schwachstellenbewertung auf allen öffentlich auffindbaren Schwachstellen im Android-Umfeld betrachtet. Mazuera-Rozo et al. betrachten in dieser Studie besonders die Art der Schwachstellen und wie sich diese über den Verlauf der Zeit ändert. Außerdem werden die Angriffsvektoren nach CVSS analysiert, die angegriffenen Ebenen und Teilsysteme von Android betrachtet und die Frage beantwortet, wie lange es dauert, bis die Schwachstellen geschlossen werden. Die Studie nutzt die MITRE CWE Kategorisierung, um die spezifischen Schwachstellen einer übergeordneten Schwachstellenkategorie zuzuordnen. Sie kommen zu dem Schluss, dass mindestens 60% der 1235 betrachteten Schwachstellen eine hohe Auswirkung auf die Vertraulichkeit(60%), Integrität(60%) und Verfügbarkeit(65%) des Geräts haben [16]. Diese Studie zeigt, wie auch die Studie von Klement, Liu und Katzenbeisser, den wissenschaftlichen Nutzen einer empirischen Analyse.

3. Theoretischer Hintergrund

In diesem Kapitel werden die technischen Hintergründe von Open RAN und O-Ran erläutert, und auf die technischen Implementierungen der 5G-FORAN-Anwendungen eingegangen. Außerdem werden die MITRE Frameworks und Datenquellen für die empirische Analyse vorgestellt.

3.1. Open RAN und O-RAN

Die Open RAN Architektur bietet eine innovative Möglichkeit, die traditionell monolithische Struktur der RAN (deutsch: Funkzugangsnetze) durch modulare und offene Ansätze zu ersetzen. Dieser Wandel wird vor allem durch die O-RAN Alliance und die O-RAN SC vorangetrieben. Ziel ist es, die Interoperabilität zwischen verschiedenen Komponentenherstellern zu verbessern und neue Möglichkeiten für Automatisierung und Sicherheit zu schaffen.

Die O-RAN Architektur, dargestellt in Abbildung 3.1, wird von der Use Cases and Overall Architecture Group (WG1) verantwortet und in verschiedenen *Task Groups* entwickelt. Sie basiert auf der Definition von standardisierten offenen Schnittstellen und einer logischen Trennung zwischen den Hauptfunktionen eines RANs. Dazu gehören unter anderem der nicht-echtzeitfähige RAN-Intelligent-Controller (Non-RT-RIC) und der nahezu-echtzeitfähige RIC (Near-RT-RIC), die jeweils für unterschiedliche Steuerungszyklen optimiert sind [17].

3.2. Virtualisierung mit Kubernetes

Die O-RAN SC Referenzimplementierung basiert grundlegend auf Kubernetes, einer Plattform zur Orchestrierung von containerisierten Anwendungen [18]. Kubernetes ermöglicht es, Anwendungen in Containern auszuführen, die von ihrer Umgebung unabhängig sind und so eine hohe Flexibilität und Skalierbarkeit bieten. Container können durch Software wie Docker erstellt werden und nutzen Virtualisierung auf Betriebssystemebene, um eine isolierte Laufzeitumgebung für O-RAN Komponenten zu schaffen. Kubernetes orchestriert diese Container, indem es deren

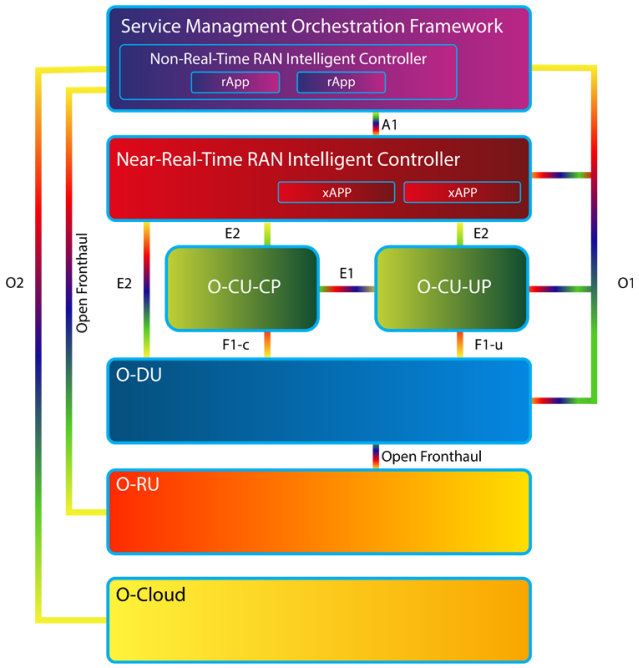

Abbildung 3.1.: Logische Architektur von O-RAN (Quelle: [10])

Bereitstellung, Skalierung und Verwaltung automatisiert. Das Konzept der Virtualisierung, auf dem Kubernetes aufbaut, hat zum Ziel, physische Ressourcen wie Prozessor und Arbeitsspeicher abstrahiert bereitzustellen. Dabei werden mehrere virtuelle Container auf einer physischen Maschine betrieben. Trotz der Vorteile bringt die Nutzung von Kubernetes und Virtualisierung auch Sicherheitsrisiken mit sich. Angriffsvektoren können durch falsch konfigurierte Container, ungesicherte Schnittstellen oder Schwachstellen in der Orchestrierungsplattform selbst entstehen [19,20].

Mit Ausnahme der O-RAN Radio Unit (O-RU) können alle Komponenten der Abbildung 3.1 vollständig containerisiert, das heißt in einer virtuellen Umgebung ausgeführt, werden. Die O-RU ist die hardware-nächste Komponente und benötigt spezialisierte Hardware zur Verarbeitung der empfangenden Signale. Die Virtualisierbarkeit der Komponenten bringt alle genannten Vor- und Nachteile in das O-RAN Umfeld ein. Insbesondere dynamische Skalierung ist im O-RAN-Umfeld entscheidend, um Netzwerkverkehr effizient zu bewältigen, wie beispielsweise von Ali und Jammal demonstriert wird [21].

3.3. 5G-FORAN

Für das ATTACK-Teilvorhaben wurde ein Framework implementiert, das es ermöglicht Angriffe auf die implementierte Open RAN Infrastruktur auszuführen. Dazu können verschiedene Tools genutzt werden, solange sie über eine Command Line Interface (CLI) verfügen. Das AT kann über diese Tools komplexe Angriffe simulieren und die Metadaten sowie Ergebnisse der Angriffe abspeichern [22]. Zur Speicherung wird eine MongoDB Datenbank genutzt. MongoDB ist eine *NoSQL*, dokumentenbasierte Datenbank [23]. Ein Dokument stellt im Kontext des ATs ein Artefakt dar, welches alle relevanten Informationen über einen Angriff enthält. Diese Datenbank ist die Basis für alle Abfragen und Visualisierungen, die über das Dashboard möglich sind.

3.4. Dashboard

Das Dashboard ist ein Visualisierungtool, über das die Gesamtmenge aller Angriffssimulationen dargestellt werden kann. Für jeden Angriff werden die Metadaten und Ergebnisse textuell aufbereitet dargestellt und mit weiteren grafischen Visualisierungen, wie einem zeitlichen Verlauf, aufgewertet. Das Dashboard ist über einen Webserver in der Programmiersprache *Go* implementiert, der serverseitig eine Darstellung der Webseite erstellt und diese beim Aufruf der spezifischen Seiten des Dashboards an den Browser des Anwenders schickt. Neben den HTML Dateien werden außerdem Cascading Style Sheets (CSS) und JavaScript (JS) Dateien ausgeliefert, welche die Weboberfläche stilisieren und Funktionalität hinzufügen, die nicht in HTML ausgedrückt werden kann. Die Architektur des Dashboards basiert auf dem Hotwire (HTML-over-the-wire) Web-Framework, welches es zum Ziel hat, möglichst wenig JS im Frontend einzusetzen. Stattdessen werden HTML Dateien dynamisch vom Server angepasst und in wenigen Millisekunden an den Client versandt. Dieser Ansatz ist weit verbreitet und wird auch als *server-sided rendering* bezeichnet [24]. Die einzelnen Seiten, die im Dashboard aufgerufen werden können, sind in einer Übersicht in Kapitel 5 aufgelistet.

Diese Arbeit umfasst die Weiterentwicklung des Dashboards auf Basis einer vorangegangenen Forschungsarbeit für die Evaluation von Dashboardtechniken und der Implementierung eines Prototypen [25].

3.5. MITRE Frameworks

Die von der MITRE Corporation (MITRE)[1] zur Verfügung gestellten Werkzeuge und Kategorisierungssysteme wurden entwickelt, um Sicherheitsrisiken zu analysieren, bewerten und behandeln. Die im Folgenden beschriebenen Frameworks bieten strukturierte Daten zur Analyse von Schwachstellen, Schwachstellenkategorien und Angriffsmustern. In diesem Kapitel werden die einzelnen Frameworks beschrieben und deren Anwendungsgebiete erläutert.

3.5.1. MITRE ATT&CK

Das MITRE Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) Framework ist ein Wissensfundus, der Angriffstaktiken und -techniken dokumentiert, die auf realen Beobachtungen basieren. Es bietet eine strukturierte Möglichkeit, Angriffsmethoden zu analysieren und deren Auswirkung auf einzelne Systeme oder Infrastrukturen zu bewerten. In der aktuellen Version 16.1 enthält das Framework 14 Taktiken und 203 Techniken in der *enterprise*-Matrix. Die Quelle ist aufgrund ihrer breiten Akzeptanz und der kontinuierlichen Aktualisierung durch die MITRE-Organisation als äußerst verlässlich einzustufen [7]. Es existieren diverse Spezialisierung der ATT&CK Matrix, die sich auf konkrete Anwendungsfälle konzentrieren, wie zum Beispiel die MITRE Adversarial Threat Landscape for Artificial-Intelligence Systems (ATLAS) Matrix. Diese beschäftigt sich insbesondere mit Angriffen auf KI-gestützte Systeme [26].

3.5.2. CVE

Das Common Vulnerabilities and Exposures (CVE)-System ist eine standardisierte Referenzdatenbank, die bekannte Schwachstellen und Sicherheitslücken in Software und Hardware dokumentiert. Jeder CVE-Eintrag enthält spezifische Informationen zu einer identifizierten Schwachstelle und bietet eine Grundlage für die Kommunikation und Priorisierung von Sicherheitsmaßnahmen, darunter die folgenden Hauptbestandteile, die für diese Arbeit besonders relevant sind:

- **CVE-ID**: Eine weltweit eindeutige Kennung im Format `CVE-JJJJ-NNNNN`, die die Schwachstelle klar identifiziert.

[1] Entgegen einem verbreiteten Missverständnis (ChatGPT-4 liegt auch falsch) steht MITRE **nicht** für "Massachusetts Institute of Technology Research and Engineering".

- **Betroffene Produkte**: Angaben zu spezifischen Softwareversionen oder Hardware, die von der Schwachstelle betroffen sind.
- **Schweregrad und CVSS-Wertung**: Integration mit dem CVSS das, neben anderen Metriken, eine Basisbewertung von 0 bis 10 bereitstellt, um die kritische Bedeutung der Schwachstelle zu quantifizieren.
- **Verknüpfungen zu CWE**: Hinweise auf zugrunde liegende Schwachstellen aus der CWE-Taxonomie.

3.5.3. CWE

Die Common Weakness Enumeration (CWE) ist eine Aufzählung von häufigen Schwachstellenkategorien und Fehlermustern in Software und Hardware, die dazu dient, Sicherheitsprobleme systematisch zu identifizieren und zu dokumentieren [27]. Jede CWE-Instanz beschreibt eine spezifische Schwachstellenkategorie, darunter die folgenden Hauptbestandteile, die für diese Arbeit besonders relevant sind:

- **Schwachstellen-ID und Name**: Eine eindeutige Kennung und Beschreibung der Schwachstellenkategorie.
- **Übliche Konsequenzen**: Eine Liste von typischen Auswirkungen, die durch Ausnutzung einer Schwachstelle dieser Kategorie verursacht werden können.
- **Beobachtete Beispiele**: Verweise auf reale Anwendungsfälle, in denen ein CVE aus dieser Kategorie ausgenutzt wurde.
- **Demonstrative Beispiele**: Codebeispiele, die die Ausnutzung einer Schwachstelle dieser Kategorie illustrieren und deren Auswirkungen verdeutlichen.

3.5.4. CAPEC

Das Common Attack Pattern Enumeration and Classification (CAPEC) Framework zielt darauf ab, Angriffsmuster systematisch zu dokumentieren und zu klassifizieren, um Sicherheitsforscher:innen und Entwickler:innen bei der Identifikation und Vermeidung von Schwachstellen zu unterstützen. Jede CAPEC-Instanz beschreibt ein spezifisches Angriffsmuster, darunter die folgenden Hauptbestandteile, die für diese Arbeit besonders relevant sind:

- **Angriffsmuster-ID und Name**: Eine eindeutige Kennung und Bezeichnung, die das Angriffsmuster klar identifiziert.

- **Beschreibung**: Eine kurze Beschreibung des Angriffs, seine Ziele und seinen Kontext.
- **Schwachstellenverknüpfungen**: Verweise auf relevante CWEs, die die zugrunde liegenden Schwachstellen beschreiben, die der Angriff ausnutzt.
- **Taxonomie-Zuordnungen**: Verweise auf Objekte des ATT&CK Frameworks

Die Datenbank ist umfassend und bietet eine hierarchische Klassifikation der Angriffsmuster, die es ermöglicht, zwischen allgemeinen Kategorien und spezifischen Instanzen zu navigieren. Diese Daten sind in der Sicherheitsforschung nützlich, da sie dabei helfen, typische Bedrohungen zu gruppieren und Szenarien für Sicherheitsüberprüfungen zu erstellen [28].

Das CVE-System unterscheidet sich von CWE dadurch, dass es konkrete Sicherheitslücken in spezifischen Produkten beschreibt, während CWE die allgemeine Kategorie oder Art der Schwachstelle adressiert. In Kombination mit der CAPEC und der CWE ermöglicht die CVE eine ganzheitliche Analyse, die sowohl Schwachstellenursachen als auch deren Ausnutzung und Abwehrmaßnahmen abdeckt. Diese Eigenschaften machen die CVE zu einem unverzichtbaren Werkzeug in der Sicherheitsanalyse und ist auch sehr gut auf Open RAN Umgebungen anwendbar [29].

3.6. Datenquellen

Die Integration von relevanten Datenquellen stellt eine wesentliche Grundlage für eine wissenschaftliche Sicherheitsanalyse dar. In diesem Kapitel werden Datenquellen beschrieben, die für diese Arbeit eine wichtige Rolle spielen.

3.6.1. Bedrohungsmatrizen

Die Threat Matrix for Kubernetes (TM4K), bereitgestellt von Microsoft, ist eine spezialisierte Threat-Matrix für Kubernetes-Umgebungen, die verschiedene Angriffsvektoren und Schwachstellen in containerisierten Plattformen kategorisiert. Sie enthält detaillierte Informationen zu typischen Angriffstechniken, Taktiken und entsprechenden Verteidigungsmaßnahmen, die speziell für Kubernetes relevant sind [30]. Die TM4K stellt damit eine Spezialisierung der MITRE ATT&CK dar, die auf den speziellen Anwendungsfall Kubernetes erstellt wurde. Die Quelle wurde seit der Erstveröffentlichung 2020 bis Anfang 2023 regelmäßig aktualisiert, hat aber seitdem keine Updates mehr bekommen, wird dennoch häufig genutzt und bietet

eine hilfreiche Wissensquelle über die Angriffstechniken im Kubernetes Umfeld [31]. Eine erweitere Version der TM4K ist die Redguard Kubernetes Threat Matrix (KTM). Sie legt einen besonderen Fokus auf spezifische Angriffe und gibt für viele Techniken Kubernetes Manifeste und CLI Kommandos an, mit denen das eigene System auf die jeweilige Schwachstelle getestet werden kann. Ihre Anwendung in der Schwachstellenanalyse von O-RAN kann hilfreich sein, da sie spezifische Ressourcen für das Finden von Schwachstellen im System enthält [32]. Die in dieser Matrix gelisteten Kommandos werden im AT genutzt, um das *Kubernetes-Testing* auszuführen. Dafür kommt kein spezielles Tool zum Einsatz, sondern die Angriffe werden direkt über die Kubernetes CLI `kubectl` ausgeführt.

Eine dritte Kategorisierung für Techniken stammt aus dem Report der O-RAN Alliance WG11. Insgesamt identifiziert diese Risikoanalyse 104 Bedrohungen, die sich gegen die Komponenten eines O-RAN Systems inklusive der O-Cloud Komponente wenden (vgl. Abbildung 3.1) [12]. Für jede dieser Bedrohungen wird eine *Threat-ID* vergeben. Aufgrund des sehr speziellen Anwendungsfalls werden in dem Report Bedrohungen beleuchtet, die nicht in der TM4K oder KTM abgebildet sind. Bisher sind die im AT implementierten Angriffe jedoch nicht so sehr spezialisiert, sodass eine Zuordnung zu einer Technik aus einer der anderen Matrizen immer möglich ist.

3.6.2. Datenquellen für empirische Analyse

In der Implementierung von ACEMA werden Daten aus externen Datenquellen genutzt. Für das Mapping von MITRE-Technik zu CVE wird das MITRE-Cyber Threat Intelligence (CTI) Repository genutzt. Weitere Informationen über einen spezifischen CVE werden über die National Vulnerability Database (NVD) des National Institute of Standards and Technology (NIST) beschafft.

Das CTI Repository enthält eine Vielzahl an Daten zur Identifikation und Kategorisierung von Cyberangriffstechniken. Es stellt spezifische Zuordnungen zwischen den im MITRE ATT&CK-Framework beschriebenen Angriffstechniken und Schwachstellen (CWEs) bereit, die eine wichtige Grundlage für Bedrohungsanalysen bilden. Diese Informationen umfassen unter anderem Angriffsmethoden, Exploit-Beschreibungen und potenzielle Abwehrmaßnahmen. Wissenschaftlich anerkannt ist das Repository durch die breite Akzeptanz des MITRE ATT&CK-Frameworks in der Sicherheitsforschung sowie durch die Anwendung in praxisorientierten Sicherheitslösungen.

Die NVD ist eine zentrale Datenbank des NISTs, die umfassende Informationen zu Schwachstellen und deren Bewertungen liefert. Sie basiert auf international

anerkannten Standards wie CVE und CVSS, um die Schwere und Ausnutzbarkeit von Sicherheitslücken zu quantifizieren. Die NVD ist aufgrund der Bereitstellung standardisierter Sicherheitsmetriken ein unverzichtbares Werkzeug in der Sicherheitsforschung. Die Kombination aus systematischer Datenaufbereitung und fundierten Bewertungen macht die NVD besonders geeignet für die Implementierung von Sicherheitslösungen und lässt sich standardmäßig in viele Management- und Dokumentationslösungen in Enterprise Umgebungen integrieren [33,34].

4. Methodik

Die Methodik dieser Arbeit beschreibt das wissenschaftliche Vorgehen bei der Implementierung des Dashboards in einer DFIR Umgebung für O-RAN. Der Fokus liegt auf der Integration einer empirischen Methode zur Bewertung von Angriffen und der Weiterentwicklung passender Visualisierungstechniken.

4.1. Theoretische Grundlagen der empirischen Methode

ACEMA ist „eine umfassende empirische Methode zur Analyse von Bedrohungen in O-RAN Umgebungen“ (eigene Übersetzung: [9]). Die Integration der Analyse-Methode von Klement, Liu und Katzenbeisser in das Forschungsprojekt 5G-FORAN bringt wertvolle Daten ein, die nützliche Visualisierungen im Dashboard ermöglichen. Im Folgenden wird erläutert, welches Ziel mit der Integration von ACEMA verfolgt wird, welchen Mehrwert ACEMA für 5G-FORAN bietet und wie die gewonnenen Daten angemessen im Dashboard visualisiert werden können.

ACEMA ermöglicht es, für spezifische MITRE-Techniken auf eine Menge an CVEs zu schließen. Das Ziel der Integration in diese Forschungsarbeit ist es, für spezifische simulierte Angriffe eine Bewertung nach dem CVSS vornehmen zu können. Die ACEMA-Arbeit hat, im Gegensatz dazu, nicht das Ziel spezifische Angriffe zu betrachten, sondern eine Übersicht über alle möglichen von der O-RAN definierten *Threat-IDs* zu geben. Der generelle Vorgang ist in der Abbildung 4.1 dargestellt.

Die spezifischen Angriffe, die im Dashboard angezeigt werden, stammen aus dem 5G-FORAN AT. Das AT implementiert aktuell circa 250 Szenarien, die Angriffsspuren erzeugen. Diese werden im Dashboard in 45 Techniken[1] sortiert und übergeordnet auf 10 Taktiken verteilt. Eine Technik kann dabei in mehreren Taktiken angewendet werden. Die Matrix im Dashboard orientiert sich stark an der TM4K, da sich

[1]Wenn ohne Zusatz von einer Technik gesprochen wird, meint das im Rahmen dieser Arbeit eine Technik aus einer der drei in Kapitel 3.6 definierten Matrizen. Nur wenn ausdrücklich von einer MITRE-Technik oder MITRE-Taktik gesprochen wird, meint dies Inhalte des ATT&CK-Frameworks.

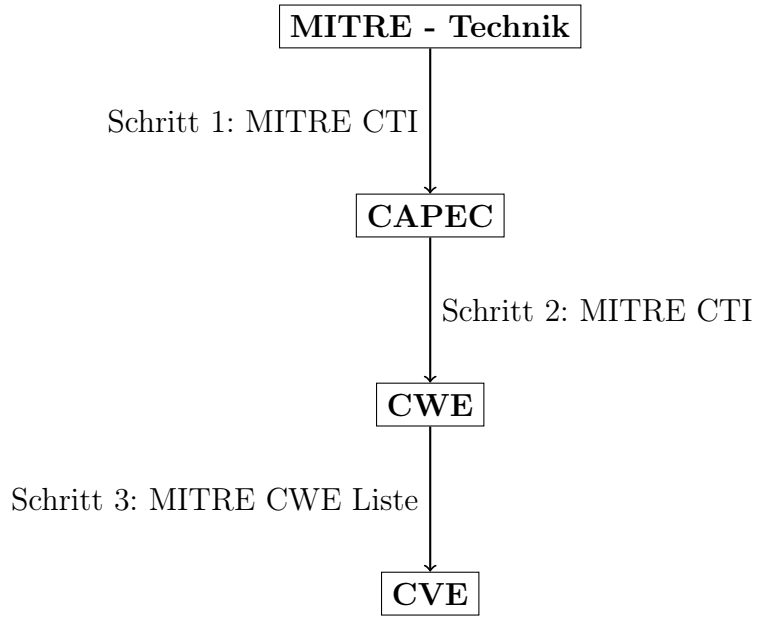

Abbildung 4.1.: Ablauf des Mappings von MITRE-Technik zu spezifischen CVE-Datum über die Kategorisierungssysteme MITRE, CAPEC, CWE und CVE.

alle Angriffe des ATs einer Technik dieser Matrix zuordnen lassen. Für über 90% der Techniken aus der TM4K ist eine Zuordnung zu einer MITRE-Technik trivial möglich.

Schritt 1 des in Abbildung 4.1 dargestellten Ablaufs sucht für eine MITRE-Technik das zugehörige Angriffsmuster, den CAPEC. Die zugrundeliegenden Daten stammen aus dem von MITRE gepflegtem CTI Repository, in welchem die Daten regelmäßig aktualisiert veröffentlicht werden [35]. Die durch ACEMA gefundenen CAPECs zu den MITRE-Techniken sind jedoch nicht vollständig, eine Erklärung und eine Lösung dafür wird in Kapitel 5.1.1 präsentiert. Schritt 2 nutzt dasselbe Repository, dort sind für das jeweilige CAPEC-Objekt auch zugehörige CWEs unter externen Referenzen verknüpft [36]. In Schritt 3 wird über das Pythonmodul *cwe2* nach zugehörigen CVEs gesucht. Die zugrundeliegenden Daten für diese Abfrage stammen aus der CWE Liste, die regelmäßig von MITRE veröffentlicht wird. Diese Liste enthält auch zugehörige CVEs, die in diesem Kontext „beobachtete Beispiele“[2] genannt werden [37,38]. Um weitere Informationen wie CVSS, Angriffsvektoren und Veröffentlichungsdatum über einzelne CVEs zu beschaffen, wird die NVD des NISTs über die Python Module `cve_lookup` und `nvdlib` abgefragt [39,40,41].

[2] Auf Englisch: „Observed Examples“

Die Besonderheit an ACEMA in Vergleich zu anderen MITRE-Technik zu CVE-Mappingtools ist, dass eine tiefgründige Analyse in die betroffenen O-RAN Bedrohungen aus dem Report der WG11 möglich ist. Dazu ist im Vorhinein eine separate Zuordnung zwischen MITRE-Technik und O-RAN *Threat-ID* aufzustellen.

Zusammenfassend müssen zwei Grundvoraussetzungen für die erfolgreiche Integration von ACEMA gegeben sein. Zum einen muss die Zuordnung einer oder mehrerer MITRE-Techniken zu einem Angriffsszenario des ATs gegeben sein. Zum anderen muss die Zuordnung einer MITRE-Technik zu einer oder mehreren O-RAN *Threat-IDs* manuell erstellt werden.

4.2. Grundprinzipien und Designentscheidungen für Visualisierungen

Die Basis für die Wahl der Visualisierungstechniken im Dashboard wurden von Jonas Weber in seiner Arbeit gelegt [25]. Weber beschreibt darin grundlegende Prinzipien, die beim Design eines Dashboards eine wichtige Rolle spielen. Über das Design von Dashboards auf Basis dieses neurologischen Konzepts gibt es zahlreiche wissenschaftliche Arbeiten, hervorgehoben sei dabei die Literaturübersicht von Barrera-Leon, Corno und De Russis [42].

Das erste Prinzip beschreibt das Phänomen der präattentiven Wahrnehmung. Darunter versteht man die Wahrnehmung von visuellen Reizen, welche jedoch unterschwellig und ohne besondere Aufmerksamkeit geschieht. Man spricht auch von "Vorbewusstsein"[43,44]. Anwendung findet dieses Prinzip in der Implementierung des Dashboards zum Beispiel beim Design der Zeitleiste. Im Listing 4.2 ist dargestellt, wie der Schweregrad von Metriken eines CVEs anhand einer farblichen Kategorisierung eingeordnet wird. Die Farbwahl beschränkt sich hierbei auf vier deutlich voneinander unterscheidbaren Farben. Die Farbzuteilung ist in Tabelle 4.1 dargestellt.

Tabelle 4.1.: Farbzuordnung zu Schweregraden

Farbe	**Schweregrad**
Grau	Keine Daten vorhanden
Grün	Niedriger Schweregrad
Gelb	Mittlerer Schweregrad
Rot	Hoher Schweregrad

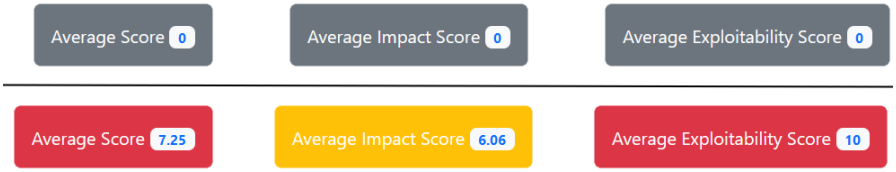

Abbildung 4.2.: Farbkategorien zur Darstellung von Metriken

Abbildung 4.3.: Hohes *data-ink* Verhältnis

Wissenschaftlich ist die kategorisierende Wahrnehmung von Farben unter anderem durch die wissenschaftliche Arbeit von Clifford et al. bewiesen [45]. Die technische Implementierung der Farbkategorisierung ist in Kapitel 5.1.2 erläutert.

Ein weiteres Prinzip, welches beim Visualisieren von Daten eine zentrale Bedeutung hat, wurde 1983 von Tufte und Katter als *data-ink*-Prinzip definiert. Das Prinzip besagt, dass der Anteil der genutzten Farbe um die Daten darzustellen möglichst hoch sein soll. Damit soll die Darstellung von nicht-daten-bezogener Ausschmückung in der Visualisierung vermieden werden [46]. An einem praktischen Beispiel kann jedoch gezeigt werden, dass ein geringeres *data-ink* Verhältnis nötig sein kann, um Daten effizient darzustellen. Als zentrale zu visualisierende Daten eines Artefakts liegen jeweils das Start- und Enddatum vor. Die Abbildung 4.3 zeigt die Darstellung von mehreren Artefakten auf einer Zeitleiste. Das *data-ink* Verhältnis ist in dieser Darstellung sehr hoch, da tatsächlich nur die Start- und Endzeitpunkte dargestellt sind. Im Vergleich dazu wird in Abbildung 4.4 die Zeitspanne zwischen den jeweiligen Daten eines Artefaktes zusätzlich farblich dargestellt. Es wird erkennbar, welche Punkte Start- oder Endpunkte sind. Dies war in Abbildung 4.3 nicht intuitiv erkenntlich. Es ist in diesem Fall sinnvoll ein geringeres *data-ink* Verhältnis zu erzielen, um Klarheit und Intuition zum Lesen zu schaffen. Die technische Konfiguration der Zeitleiste ist in Kapitel 5.3 erläutert.

Diese beiden Prinzipien und andere Best Practices wurden auch bei Weiter- und Neuentwicklungen von Visualisierung im Dashboard gewahrt. Bei der Darstellung des Angriffspfads und der Netzwerktopologie wurden bewusste design-technische Entscheidungen getroffen. Die Angriffspfadsvisualisierung enthält nicht viele Daten,

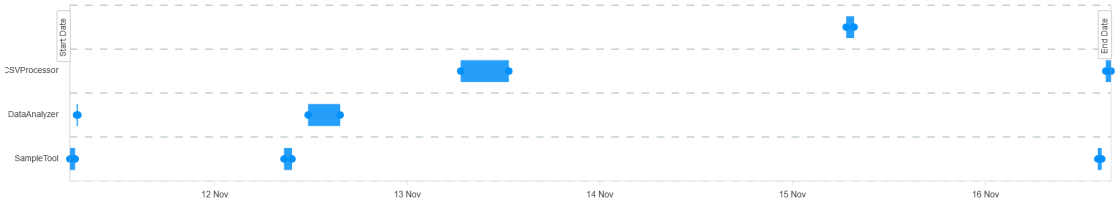

Abbildung 4.4.: Niedriges *data-ink* Verhältnis

nimmt aber einen großen Platz auf der Detailübersicht eines Artefaktes ein. Ein gerichteter Graph bietet sich als Visualisierungstechnik an, da dieser leicht zu verstehen ist. Jedes Element im Graphen besitzt auch nur maximal eine Kante, die auf das nächste Element in der Kette des Angriffspfades steht. Der Angriffsgraph ist beim Laden der Seite nicht direkt, sondern nur nach einem bewussten Scrollen sichtbar. So wird der Nutzer[3] nicht mit zu vielen Informationen auf einmal überladen. Diese Entscheidung wird unterstützt durch die Studie von Toreini und Langner, in der die Aufnahmefähigkeit in Verbindung mit der Anzahl von gleichzeitig sichtbaren Visualisierungen untersucht wird [47].

Die Graphen werden nicht als statische Bilder eingebettet. So lassen sich weitere Funktionen wie Tooltips, dynamischer Zoom und die Neuanordnung der einzelnen Elemente durch den Nutzer realisieren. Ein Tooltip hat die Funktion, zusätzliche hilfreiche Information zu liefern, wenn der Nutzer mit dem Mauszeiger auf ein Element zeigt [48]. Die Darstellung eines Tooltips ist in Abbildung 4.5 gezeigt. Über die Verschiebbarkeit der Elemente kann das Diagramm angepasst werden, um es verständlicher darzustellen oder für einen Export vorzubereiten. Die Graphen können über gängige Browser per *Rechtsklick, Speichern unter / In neuem Tab öffnen* als `.png` Datei exportiert werden. Die technische Implementierung des Angriffsgraphs, vom als Text definierten Graphen bis zur Visualisierung im Dashboard, und das Ergebnis der Implementierung sind in Kapitel 5.2 erläutert.

4.3. Werkzeuge und Technologien

In diesem Kapitel wird etwas tiefer auf die Tools eingegangen, die zur Umsetzung der Implementierung des Dashboards genutzt wurden. Informationen über die generelle Architektur, genutzte Programmiersprachen und Technologien finden sich in Kapitel 3.4.

[3] In dieser Arbeit wird aus Gründen der besseren Lesbarkeit das generische Maskulinum für eine Person verwendet, die das Dashboard nutzt. Alle Geschlechteridentitäten werden dabei ausdrücklich mitgemeint.

Abbildung 4.5.: Tooltip auf einem Element im Angriffsgraph

Zu Beginn wurde die Entwicklung auf einer rein lokalen Umgebung durchgeführt. Über das von MongoDB zur Verfügung gestellte CLI lief ein MongoDB-Server, ein per `go build` gebauter Dashboard-Server und der per `yarn dev` gestartete Vite-Entwicklungs-Server auf dem lokalen Rechner [23,49]. Die Testdaten stammten aus dem Gitlab-Repository und wurden vom vorherigen Entwickler zur Verfügung gestellt [50]. Die Qualität und Menge der Testdaten waren für den Anfang ausreichend. Der Testdatensatz wurde seitdem aktualisiert, um den aktuellen Stand der Daten zu repräsentieren, und kann über die MongoDB CLI mit `mongoimport` geladen werden [51]. Um mit aktuellen Daten zu arbeiten und näher an der etablierten Testumgebung zu sein, wurde ab dem 7. August 2024 eine virtuelle Maschine (VM) in der Umgebung des Labors für Datennetze (DN.Lab) der TH Köln genutzt. Die Anwendung konnte nun entweder weiterhin lokal auf der VM oder zentralisiert per Ansible bereitgestellt werden [52]. Ansible ist ein Tool zur Orchestrierung von Systemen und Software, welches durch die Ausführung von Skripten einen spezifischen technischen Zustand herstellt [53]. Für die Ausführung des ACEMA-Quellcodes ist das Aufsetzen einer Python-Umgebung nötig [54,55]. Bei der Implementierung des Dashboards wurde für die Entwicklung und das Debugging gelegentlich ChatGPT-4o von OpenAI genutzt, um allgemeine Programmierfragen zu klären und Unterstützung bei Bearbeitung von repetitiven Aufgaben zu erhalten.

5. Implementierung

In diesem Kapitel wird die praktische Umsetzung der im Methodikteil beschriebenen Ansätze detailliert erläutert. Der Schwerpunkt liegt dabei auf der Anwendung und Integration der empirischen Methode ACEMA. Außerdem wird beschrieben, wie und wo die Integration von CVSS Daten vorgenommen wurde. Weiterhin wird auf die Visualisierung des Angriffspfads eingegangen und erklärt, welche Tools dafür eingesetzt wurden. Abschließend werden zusätzliche Implementierungen vorgestellt, welche die Funktionalität und Benutzerfreundlichkeit des Dashboards erweitern, darunter die Verbesserung der Zeitleistenansicht, die Vereinfachung der Datenstruktur von Artefakten sowie weiterführende Informationen zu genutzten Techniken.

Grundlegend besitzt das Dashboard vier verschiedene Seiten, die aus dem Hauptmenü angesteuert werden können. Die Seiten von Artefakten und Kampagnen sind dabei in die Übersichtsseite und Detailsansicht zu trennen, ACEMA und Matrix besitzen nur die Hauptroute. In der Abbildung 5.1 sind alle Seiten dargestellt, die direkt vom Dashboard aus erreichbar sind. In der Liste in Anhang E werden für alle Seiten im Dashboard detailierte Informationen bereitgestellt. Diese umfassen eine allgemeine Beschreibung der Funktion, mögliche Filtermöglichkeiten der Daten und es wird erklärt welche Daten dazu beitragen die Seite dynamisch mit Inhalt zu füllen.

5.1. Implementierung von ACEMA

Die empirische Methode kann auf zwei verschiedene Arten für das Dashboard genutzt werden. Zum einen lassen sich über die Matrixdaten aus Anhang D sehr aussagekräftige Diagramme erstellen, die jedoch nicht direkt in die Technik- oder Artefaktdarstellung im Dashboard integriert werden. Zum anderen gibt der Output von ACEMA die Möglichkeit, die CVSS Daten direkt in existierende Seiten des Dashboards zu integieren und den Inhalt aufzuwerten.

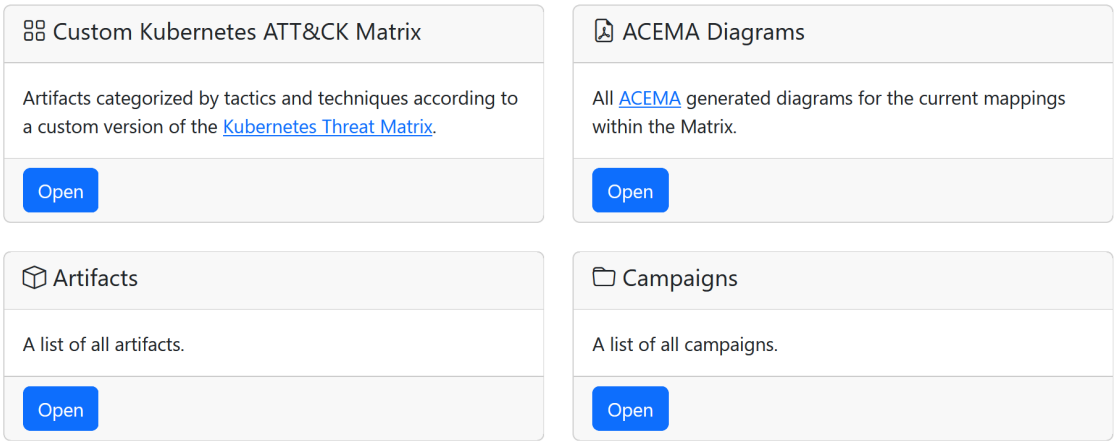

Abbildung 5.1.: Seiten, die von der Startseite des Dashboards aus geöffnet werden können.

```
    AddEntryToMap(
        "MS-TA9002",                        // KEY in die Map
        "MS-TA9002",                        // TM4K-ID
        []string{"T1195.002", "T1525"},     // ATT&CK-Technik-IDs
        "Compromised Image In Registry",    // Technik Name
        []string{"initial-access"},         // ATT&CK-Taktik-Namen
        []string{"T-IMG-01"},               // O-RAN Threat-IDs
    )
```

Listing 5.1: Beispielhafte Definition einer Technik im Dashboard

5.1.1. Anwendung von ACEMA auf die Daten

Der allgemeine Ablauf der Mappings wurde in Kapitel 4.1 bereits angeschnitten. Es wurden zwei Grundvoraussetzungen definiert, die für die erfolgreiche Implementierung von ACEMA nötig sind. Die erste Voraussetzung (Zuordnung Angriffsszenario zu MITRE-Technik) wurde von den Verantwortlichen des ATs umgesetzt. Die zweite Voraussetzung bedurfte einer manuellen Zuordnung von Techniken zu O-RAN *Threat-IDs*. Diese Zuordnung ist einmal manuell nötig und konnte aufgrund der ausführlichen Beschreibungen der Techniken und *Threat-IDs* ohne große Aufwände hergestellt werden. Diese Zuordnung ist in die Datenstruktur einer Technik im Dashboard integriert. Eine Technik in der modifizierten Matrix wird wie in Listing 5.1 beispielhaft gezeigt definiert.

Ein Export im `csv` Format aller definierten Techniken ist in Anhang D abgebildet. Dieser Datensatz stellt auch gleichzeitig die Eingangsdatei dar, auf dessen Basis ACEMA arbeitet. Der Datenexport wird erzeugt, wenn die Dashboardanwendung gestartet wird.

Die Implementierung von ACEMA lässt sich in zwei Teile spalten: den `Gathering`-Teil (deutsch: Sammlungsteil) und den `Analysis`-Teil (deutsch: Analyseteil). Es existieren hierfür jeweils eine Datei mit den aufrufbaren Funktionen und eine andere, die die Aufrufe dieser Funktionen tätigt und prozedural ausgeführt wird.

Es wurden einige Anpassungen des verfügbaren ACEMA-Quellcodes vorgenommen, die den Entwicklungsfluss vereinfachen und Verbesserungen vornehmen. Eine vorgenommene Verbesserung bezieht sich auf die Vollständigkeit des Daten-Mappings. Es wird in Kapitel 6.2 gezeigt, dass die im Folgenden beschriebene Verbesserung zu der Schöpfung einer größeren Datenmenge aus dem CTI Repository führt. Eine weitere Änderung betrifft die Ausführung der Python Skripte. Die prozeduralen Teile sind im originalen Repository als `jupyter notebooks` (`.ipynb`-Dateien) geschrieben [54,56]. Diese Dateien wurden zu einfacheren Ausführung in `.py` Dateien konvertiert, sodass sie direkt über die Python-CLI aufrufbar sind. Eine Ausführung in einem Jupyter Notebook ergibt für das erstmalige Nachvollziehen oder Demonstrationszwecke Sinn, ist aber für die spätere Ausführbarkeit hinderlich.

Informationen über alle genutzten Quellcodes in dieser Arbeit und wo diese zu finden sind, werden im Anhang A zur Verfügung gestellt.

Der originale Quellcode arbeitet beim Mapping von MITRE-Technik zu CAPEC mit einem *pandas dataframe*. Pandas ist eine bekannte Datenanalysebibliothek für Python [57]. Für das Konvertieren von Structured Threat Information Expression (STIX)-Daten zu einem *dataframe* wird die Funktion `mitreattack.stixToDf` genutzt [58,59]. Es wird dadurch der *enterprise-attack*-Teil der CTI Daten durchsucht. Viele Beziehungen zwischen MITRE-Technik und CAPEC finden sich jedoch auch im *capec/2.1/attack-pattern* Teil des Repositories. Diese Limitation bringt signifikant weniger Mappings zum Vorschein, wie empirisch in Kapitel 6.2.2 gezeigt wird. Deshalb nutzt die verbesserte Implementierung die Datensätze aus beiden Teilen. Auch wurde ein anderer Ansatz zur Verarbeitung der einzelnen `json`-Dateien angewandt. JSON ein standardisiertes Format für strukturierte Daten. Anstatt der Konvertierung zu einem mächtigen *dataframe* der alle Daten enthält, werden die STIX-formatierten Daten aus jeder `json`-Datei einzeln geladen und der Inhalt der STIX-Objekte über `stix_data.get()` ausgelesen [60]. Die genaue Abfolge der abgefragten Kriterien bis zum Finden eines Mappings in Abbildung 5.2 ist als detaillierte Ansicht des Schrittes 1 in Abbildung 4.1 zu verstehen.

Wenn alle Mappings über die verbesserte Implementierung gefunden wurden, werden die Ergebnisse mit den Ergebnissen aus der originalen Implementierung zusammengeführt. Dadurch ergibt sich das komplette Mapping von MITRE-Technik zu CAPEC. Für die anderen Schritte des Mappings wurden keine Verbesserungsmöglichkeiten gefunden, daher sind die Funktionen aus dem originalen Quellcode

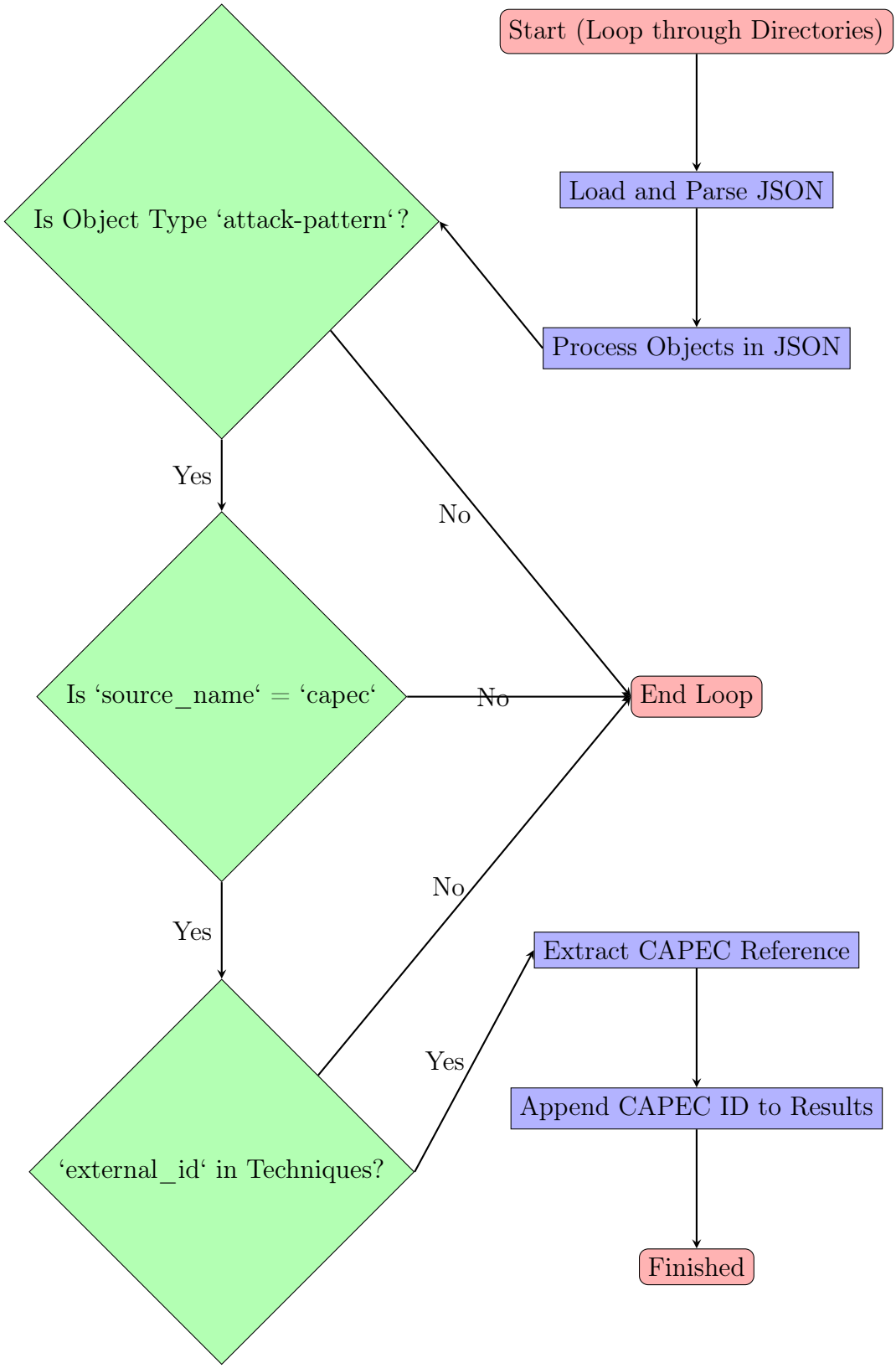

Abbildung 5.2.: Detailansicht des Ablaufs des Mappings von MITRE-Technik zu CAPEC.

beibehalten worden. Die Ausgabe der ACEMA Ausführung ist eine Datei im `json` Format. Ein beispielhafter Auszug aus dieser Datei ist im Anhang B dargestellt. Bis zur Erstellung dieser Datei vergingen beim letzten Durchlauf knapp unter 40 Minuten. Das ist der hohen Anzahl der CVEs geschuldet, für die jeweils eine Anfrage an die NVD geschickt wird. Die Dokumentation der NVD Schnittstelle empfiehlt eine Wartezeit von sechs Sekunden zwischen Anfragen [61].

$$353 \text{ CVEs} \cdot 6 \text{ Sekunden Wartezeit} \approx 35min$$

Der Output des *Gathering* Skripts wird für die Analyse der Daten verwendet, um verschiedene Arten von Diagrammen zu erzeugen. Auf die Erkenntnisse, die man aus diesen Diagrammen gewinnen kann, wird in Kapitel 6.1.1 eingegangen. Alle Diagramme können auch im Dashboard angezeigt werden. Dazu wurde die Seite `/acema` erstellt, die alle PDF-Dateien aus dem Ordner `public\data` enthält. Neben den von Klement, Liu und Katzenbeisser erstellten Diagrammen wurden einige weitere implementiert, die tiefere Einsicht in die Daten geben. Die Erstellung zusätzlicher Diagramme wird durch die Nutzung bekannter Datenanalyseframeworks und die umfangreiche Dokumentation im Quellcode unterstützt.

Aktuell ist es nicht implementiert, die ACEMA Python Skripte aus dem Dashboard heraus auszuführen, um eine neue Datensammlung oder Datenauswertung zu starten. Die Änderungsrate der Daten ist zudem eher gering, insofern ist eine Aktualisierung in kleinen Abständen nicht nötig. Die MITRE-Techniken im Dashboard sind quasi statisch, da sie sich an den implementieren Szenarien des ATs und ATT&CK-Frameworks in Version *16.1* orientieren. Insgesamt umfasst die Gesamtmenge aller CAPECs 559 in CAPEC Version *3.9*, wobei seit Januar 2023 nur vier neue (+0,7%) hinzugefügt wurden. Auch die Veränderung im Mapping zwischen CAPEC und CWE ist gut dokumentiert auf der CAPEC Webseite einsehbar. Hier ist eine Veränderung von 59 neu hinzugefügten Mappings in Version *3.9* vor dem Hintergrund der Gesamtmenge auch eine sehr kleine Veränderung. Die Gesamtmenge der CAPEC-zu-CWE-Mappings lässt sich anhand folgender Daten aus dem Testdatensatz in Anhang D und dem Wert aus [62] schätzen:

$$\begin{aligned}
&\text{Anzahl von CAPEC in Testdatensatz} && = 29 \\
&\text{Anzahl von CWE in Testdatensatz} && = 119 \\
&\text{Durchschnittliche Anzahl von } \frac{\text{CWEs}}{\text{CAPEC}} && = 4{,}1 \\
&\text{Gesamtmenge aller CAPECs} && = 559 \\
&\text{Schätzung der Anzahl der Mappings} && = 559 \cdot 4{,}1 \\
& && \approx 2300
\end{aligned}$$

Zu einem ähnlichen Wert von 1700 bis 2800 kommt auch OpenAI ChatGPT 4o, wobei dort mit Werten von 3 bis 5 für die Anzahl der durchschnittlichen CWEs pro CAPEC gerechnet wird [63].

Zuletzt muss noch das Mapping von CWE zu CVE betrachtet werden. Die Daten dazu stammen aus der MITRE CWE Liste und werden mehrmals jährlich, ungefähr alle vier Monate aktualisiert [38,64].

Das Mapping zwischen MITRE-Technik und CVEs ändert sich infolgedessen nur in geringer Weise. Eine Integration der neuen Daten durch das Ausführen der ACEMA Skript und das Kopieren der `json`-Datei in die Dashboardverzeichnisse `pkg\artifacts` und `pkg\mitre` ist demnach nicht kontinuierlich, aber mit einem Abstand von ungefähr vier Monaten, angepasst an die Änderungsrate der CWE Liste, empfehlenswert.

5.1.2. CVSS Integration

Eine Integration mit dem CVSS ist eine grundlegende Funktion für jede Anwendung, die eine Bewertung von Schwachstellen vornimmt. Der CVSS Wert hilft dabei, schnell einen Überblick über den Schweregrad eines potenziellen Angriffs durch Ausnutzung der spezifischen Schwachstelle zubekommen. Aus diesem Grund hatte die Umsetzung von Beginn an eine hohe Priorität. Ursprünglich sollte die Bewertung nur über manuell zugeordnete CVEs vorgenommen werden, dieser Ansatz stellte sich allerdings als mühsam und nicht praktikabel heraus. Über die in Kapitel 5.1.1 beschriebene Implementierung von ACEMA war eine deutlich schnellere und automatisierte Anreicherung von Daten mit CVEs möglich. Die Visualisierung von spezifischen Werten, die sich aus insgesamt sechs CVSS Metriken zusammensetzen, wurde an zwei Stellen implementiert, die im Folgenden genauer betrachtet werden. Der in Listing 5.2 dargestellte und im nachfolgenden Kapitel beschriebene Go Quellcode arbeiten nicht direkt mit dem originalen Output von ACEMA (Dateiname: `t-cwe-cve-dict.json`), sondern mit einer eigens leicht abgewandelten Version, in der nur die für das Dashboard relevanten Daten vorhanden sind. Diese *neue* Datei mit dem Suffix *_small* wird über das Ausführen von `data_analysis_nb.py` erstellt. Darin sind nur die Daten enthalten, die für die Integration einer Bewertung nach CVSS nötig sind, wie in Anhang C zu erkennen ist. Die enthaltenen Daten beschränken sich auf MITRE Technik-ID, CAPEC-ID, CWE-ID, CVE-ID, und vier Metriken aus zwei Metrikgruppen die in CVSSVersion 2 definiert sind [65]. Weiterführende Information über Metrikgruppen finden sich im Folgenden.

Um die Outputdatei mit den relevanten Daten im Dashboard nutzen zu können, wird das Paket `embed` aus der Standard Bibliothek von Go genutzt [66]. Dies

```
1   //go:embed t-cwe-cve-dict-small.json
2   var data_json embed.FS
3   if ACEMA_DATA.Data == nil {
4       fileData, err := data_json.ReadFile("t-cwe-cve-dict-small.json")
5       if err != nil {
6           log.Fatal(err)
7       }
8       err = json.Unmarshal(fileData, &ACEMA_DATA.Data)
9       if err != nil {
10          log.Fatal(err)
11      }
12  }
```

Listing 5.2: Datei in Binardatei einbetten und in Struktur überführen

ermöglicht es, auf die Datei zuzugreifen, nachdem das Dashboard Projekt mit `go build cmd/main.go` zu einer Binärdatei kompiliert wurde. Der Quellcode in Listing 5.2 zeigt, wie die Datei in eine Variable gelesen wird, die ein lokales Dateisystem simuliert. Von dort kann die Datei in die vorher definierte Liste vom Datentyp `ACEMA_DATA` gelesen werden. Die Variable `ACEMA_DATA.Data` agiert hierbei quasi als *Immutable Object* (deutsch: unveränderliches Objekt), das heißt der Wert einer Variable wird während der Laufzeit einmal gesetzt und nicht mehr verändert. Die Sprache Go hat keinen eingebauten unveränderlichen Datentyp, der bei Laufzeit befüllt werden kann. Außerdem wird die Datei nicht unnötig aus dem Speicher gelesen, wenn die Daten bereits in der Variable vorhanden sind. Die Datenstruktur folgt daher dem Write-Once, Read-Many (WORM)-Prinzip. Ausgelesen werden die Daten mehrfach während der Visualisierung von CVSS Daten.

Die Visualisierung von einem durchschnittlichen[1] CVSS Wert für eine TM4K-Technik in der Matrix ist einer der Anwendungszwecke, die durch das von ACEMA erstellte Mapping möglich gemacht wird. Dabei wird für eine Technik der Durchschnitt aller CVSS *v2_ score* Werte in den zugeordneten CVEs gebildet (siehe Zeile 1 im Listing 5.3). Wie bereits in Kapitel 6.2.2 beschrieben, nutzt ACEMA das CVSS in Version 2. Die Metrik *v2_ score* ist dabei ein Wert, der sich aus allen vorhandenen Metriken aus den verschiedenen Metrikgruppen zusammensetzt. In der *Gruppe der Basismetriken* werden die grundlegenden Merkmale einer Schwachstelle bewertet, die sich, im Gegensatz zu Metriken aus der *zeitlichen Gruppe* und der *Umgebungsgruppe*, nicht über die Zeit verändern [65].

Bei der Ausführung von `data_analysis_nb.py` wird auch für jede MITRE-Technik die Berechnung der durchschnittlichen CVSS Werte über alle gefundenen CVEs durchgeführt. In der Funktion `generate_json_with_scores` werden dazu die jeweiligen Werte in einer Liste gesammelt und mithilfe der Funktion `mean(list[])`

[1] In diesem Kapitel wird über den Durchschnitt von Werten gesprochen, gemeint ist immer das arithmetische Mittel.

```
1     new_technique["avg_score"] = statistics.mean(all_v2_scores)
2     new_technique["avg_impact_score"] = statistics.mean(all_v2_impact_scores)
3     new_technique["avg_exploitability_score"] = statistics.mean(all_v2_exploitability_score)
```

Listing 5.3: Berechnung des arithmetischen Mittels aus den CVSS Metriken mehrerer CVEs

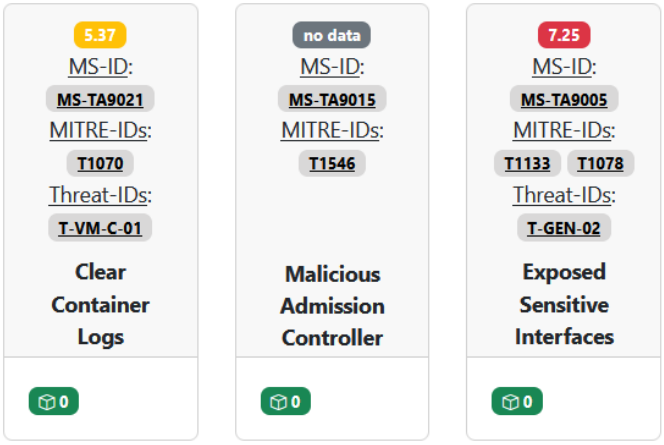

Abbildung 5.3.: Visualisierung des Basiswerts für eine Technik

aus dem Python Modul `statistics` das arithmetische Mittel berechnet, wie in Listing 5.3 gezeigt.

Die arithmetischen Mittel der Metriken werden daher somit direkt in die `json`-Datei geschrieben und müssen nicht bei einem Neustart des Dashboards neu berechnet werden.

Eine visuelle Darstellung der Berechnung des *Basisscores* ist in Abbildung 5.4 gezeigt. Zu ungefähr 90% der im Dashboard dargestellten Techniken ist eine MITRE-Technik zugeordnet und über ACEMA können für diese insgesamt 306 relevante CVEs gefunden werden. Diese Menge an CVEs bietet eine ausreichende wissenschaftliche Basis zur Auswertung von CVSS Daten. Es sind keine zu visualisierende Daten verfügbar, wenn entweder keine CVEs zu der MITRE-Technik gefunden werden oder keine MITRE-Technik zu der Technik im Dashboard zugewiesen ist. Drei Beispiele sind in Abbildung 5.3 abgebildet. Die gesamte Matrix mit allen Zuordnungen und CVSS Werten ist im Anhang G dargestellt.

Eine weitere Anwendung findet die Integration mit CVSS Daten in der Übersichtsseite eines Artefakts. Die dort angezeigten Werte sind auch wieder Durchschnittswerte, die sich aus den CVEs zusammensetzen, die genau der einen MITRE-Technik in

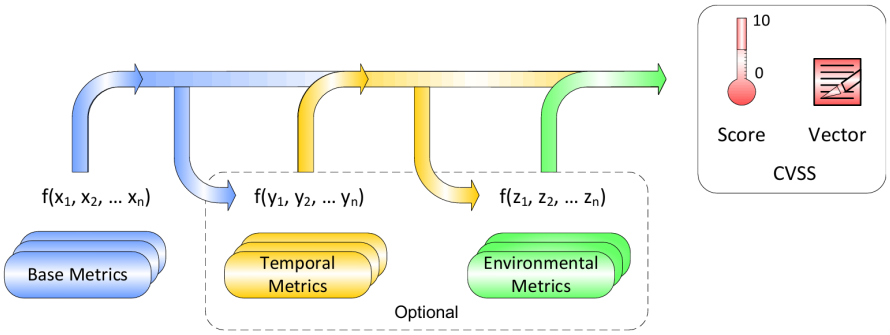

Abbildung 5.4.: Einflüsse in der Berechnung des CVSS *Basisscores* über die drei Metrikgruppen

diesem Artefakt zugeordnet wurden. Zusätzlich zu dem generellen Durchschnittswert werden in der Detailansicht zwei weitere Wert dargestellt, die die Auswirkung und Ausnutzbarkeit der zugeordneten CVEs darstellen. Einer der Werte setzt sich aus drei der Metriken aus der *Gruppe der Basismetriken* zusammen. Alle *Impact* (deutsch: Auswirkung) Metriken werden zu einem *v2_ impact_ score* zusammengefasst. Der andere Wert stammt aus der zeitlich veränderlichen Metrikgruppe und beschreibt die *Exploitability* (deutsch: Ausnutzbarkeit) der Schwachstelle [65]. Die Bewertung der Ausnutzbarkeit verändert sich zum Beispiel, wenn ein Update für die angreifbare Software veröffentlicht wird, das die Sicherheitslücke schließt. Über die Darstellung der Werte wird eine gute Übersicht der Schwere einer Schwachstelle gegeben. Nach dem Basisscores sind die Angaben der Bewertung der Ausnutzbarkeit und Auswirkungen die gängigsten Metriken. Die Berechnung dieser Werte wird durch die folgenden Formeln aus der CVSS-Spezifikation ausgedrückt.

$$\begin{aligned}\text{Auswirkungen} = 10{,}41 \cdot \Big(1 - (1 - \text{Vertraulichkeitsverlust}) \\ \cdot (1 - \text{Integritätsverlust}) \cdot (1 - \text{Verfügbarkeitsverlust})\Big)\end{aligned}$$

$$\begin{aligned}\text{Ausnutzbarkeit} = 20 \cdot \text{Angriffsvektor} \cdot \text{Angriffskomplexität} \\ \cdot \text{Authentifizierung}\end{aligned}$$

$$f(\text{Auswirkungen}) = \begin{cases} 0, & \text{wenn Auswirkungen} = 0, \\ 1{,}176, & \text{sonst.} \end{cases}$$

$$\begin{aligned}\text{Basisscore} = \text{Runden auf 1 Dezimalstelle}\Big[\Big((0{,}6 \cdot \text{Auswirkungen}) \\ + (0{,}4 \cdot \text{Ausnutzbarkeit}) - 1{,}5\Big) \cdot f(\text{Auswirkungen})\Big]\end{aligned}$$

In beiden Fälle ergänzt eine farbliche Darstellung die numerischen Werte. Die Implementierung im Go Quellcode ist in Listing 5.4 gezeigt. Die Schwellwerte, die ein Spektrum zu einer diskreten Farbe abbilden, können hier angepasst werden. Diese Funktion wird dann über die `template.FuncMap` als Funktion definiert, die direkt in einer `.tmpl` Datei genutzt werden kann. *Go Templates* (deutsch: Vorlagen) werden in Go genutzt, um datengesteuert HTML-Seiten mit dynamischen Daten aus einer Datenquelle zu befüllen. Die Datenquellen sind in der Implementierung des Dashboards Objekte oder eine Liste von Objekten vom Datentyp `Tactic`, `Technique` oder `Artifact`. Die Möglichkeit, eine Funktion über `template.FuncMap` verfügbar zu machen, erweitert die Standardfunktionen des Template-Werkzeugs und erlaubt es, komplexe logische Operationen oder Formatierungen direkt im Template auszuführen. Dies führt zu einer klaren Trennung von Logik in *.go* und Darstellung in *.tmpl* Dateien [67].

Die Darstellung von einzelnen GUI-Elementen wird durch die Nutzung von Bootstrap-Komponenten unterstützt. Bootstrap bietet unter anderem vorgefertigte *Badges* (deutsch: Plaketten) an, die genutzt werden, um zum Beispiel den CVSS Wert für eine Technik in der Matrix darzustellen. Die Nutzung von *Badges* ist in Abbildung 4.2, die zugehörige Zeile des Quellcodes in einer Template-Datei ist in Listing 5.5 gezeigt [68].

```
func ScoreColor(score float64) string {
        switch {
                case score > 7.0:
                return "danger"
                case score > 3.0:
                return "warning"
                case score == 0.0:
                return "secondary"
                default:
                return "success"
            }
    }
```

Listing 5.4: Implementierung der farblichen Kategorisierung von Schweregraden

```
<span class="badge bg-{{ scoreColor .AvgV2Score }}">{{ .AvgV2Score }}</span>
```

Listing 5.5: Aufbau eines HTML Elements zur Darstellung eines CVSS Werts in einer Bootstrap Komponente

Die Rückgabe der Funktion `ScoreColor` wird im Template direkt als String in die Definition einer *Badge* eingefügt. So lassen sich vorgefertigte Farben anhand von damit assoziierten Adjektiven (z.B.: success = grün) auswählen. Am Beispiel von Listing 5.5 wird erkennbar, dass die Funktion `scorecolor` mit dem Attribut `AvgV2Score` eines Artefakts als Parameter aufgerufen wird, dessen Rückgabe die Klasse des HTML-Elements `span` definiert. Der tatsächliche Wert des Attributes wird dann innerhalb des Elements ausgegeben [69].

Eine Verwendung weiterer Daten, wie weiterführende Informationen über spezifische CVEs und CWEs, die in der originalen `t-cwe-cve-dict.json` Datei vorhanden sind, ist in der aktuellen Version des Dashboards nicht angedacht und daher nicht implementiert.

5.2. Visualisierung des Angriffspfads

Die Visualisierung des Angriffspfads bietet im Dashboard eine intuitive Darstellung der Angriffssequenzen in Form eines Netzwerks, das durch Icons ergänzt wird. Der Begriff *Netzwerk* meint in diesem Kontext einen gerichteten Graphen (Digraph), also ein Diagramm mit Knoten und Kanten, in dem die Kanten durch einen Pfeil dargestellt werden, der von einem Knoten auf einen Anderen zeigt [70]. Die Darstellung erlaubt eine schnelle Erfassung der kritischen Informationen, indem sie zeigt, welches Gerät mit welchem Software-Werkzeug und welchem ausgeführten Kommando in Verbindung steht, um die Angriffssimulation auszuführen. In der

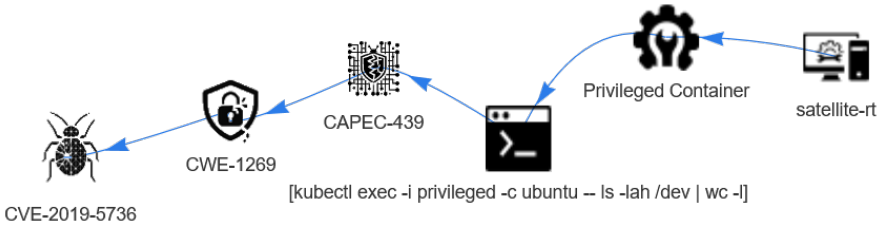

Abbildung 5.5.: Visualisierung eines Angriffspfads

Verkettung werden außerdem die Daten über das genutzte Angriffsmuster (CAPEC), Schwachstellenkategorie (CWE) und letztendlich die spezifische Schwachstelle (CVE) visualisiert. Das Ergebnis ist in Abbildung 5.5 veranschaulicht.

Ein Teil der Informationen stammt direkt aus dem Angriffstool und wird aus der Datenbank gelesen. Dazu gehört zum einen der Gerätename, auf dem das AT ausgeführt wird. Wenn ein Angriff über die CLI eines Tools ausgeführt wird, ist der Name dieses Tools in der Datenbank hinterlegt, sowie das Kommando, das die Ausführung des Angriffes startet. Diese Informationen müssen nicht zwangweise hinterlegt sein, sind aber in den meisten Fällen vorhanden. Über den Wert des Feldes `tool_name` kann in der Übersicht aller Artefakte gefiltert werden.

Ein anderer Teil wird aus den ACEMA Daten zugeordnet. Alle relevanten Daten werden, wie in Listing 5.2 gezeigt, in der Funktion `InitACEMA()`in eine Liste von Objekten vom Typ `ACEMA_DATA` geladen. Wenn dem Artefakt über die Datenbank eine MITRE-Technik zugeordnet wurde, werden diese Daten nach dieser `technique_id` durchsucht und bei dem *ersten Match* CAPEC und CWE zugeordnet. Diese Logik ist in der Funktion `AddACEMA()` zu finden. Im ACEMA-Datensatz, der aktuell im Dashboard verwendet wird, ist es in weniger als 15% der Fälle so, dass mehrere CAPECs zu einer MITRETechnik gefunden werden. Häufiger (vgl. Schätzung aus Kapitel 5.1.1) kommt es vor, dass einem CAPEC mehrere CWEs zugeordnet sind. Da nach dem Prinzip *erster Match* die Daten für die Angriffspfadvisualisierung gesucht werden, ist diese Zuordnung mit Vorsicht zu genießen. Wichtig zu beachten ist, dass keine spezifischen CVEs zugeordnet werden, da hier die Fehlerrate über den *ersten Match* noch größer wäre. Über ACEMA wird eine generelle Menge an möglichen CVEs gefunden. Welcher spezifische CVE letztendlich von dem jeweiligen Tool ausgenutzt wird, kann darüber nicht klar zugeordnet werden. Um eine genaue Zuordnung herzustellen, ist ein manueller Eintrag vom AT in das Feld `cve` des jeweiligen Artefakts nötig. Für die Daten CAPEC und CWE wird dieses Risiko der möglicherweise ungenauen Zuordnung eingegangen, da es sich nicht um hochkritische Daten handelt. Zwischen CAPECs und CWEs die einer MITRE-Technik zugeordnet sind, ist immer eine deutliche Ähnlichkeit zu erkennen.

Für die textbasierte Erstellung von Diagrammen wird in der Wissenschaft oft die quell-offene Software *GraphViz* und ihre standardisierte Directed acyclic graph of tommorow (DOT) Skriptsprache verwendet. Mithilfe dieser Tools ist es möglich, auf eine flexible und klare Weise Diagramme textuell zu definieren und in wissenschaftliche Arbeitsschritte zu integrieren [71,72]. Im Dashboard wurde die Erstellung von Graphen im DOT Format genutzt, da sich ein einfacher Graph innerhalb weniger Zeilen ausdrücken lässt und es JS Pakete gibt, die die Umwandlung von Text zu Graph komfortabel zulassen.

Die Erstellung eines DOT-Graphen in HTML erfolgt in mehreren Schritten, um eine dynamische und visuelle Darstellung zu ermöglichen. Voraussetzung ist, dass die zu visualisierenden Daten des Artefakts in der Datenbank existieren. Außerdem ist die Vorlage des Graphens als ein Go String definiert. Diese Vorlage dient als Grundgerüst für den Graphen und enthält Platzhalter, die später durch spezifische Daten gefüllt werden. Die Struktur des Templates ist so gestaltet, dass sie die grundlegenden Elemente eines DOT-Graphen, wie Knoten, Kanten und Attribute, klar definiert. In Abbildung 5.6 ist der Prozess visuell dargestellt. Der Fluss von Daten ist dort mit Pfeilen dargestellt, die Blöcke beschreiben durchgeführte Aktionen in der jeweiligen Umgebung.

Der Prozess beginnt mit dem Auslesen der Daten aus der Datenbank, sobald die Detailseite eines Artefakts aufgerufen wird. Bei jedem Aufruf der Seite wird der Graph neu erstellt, um auf aktualisierte Daten reagieren zu können. Die Artefaktdaten existieren jetzt als ein Objekt in Go und können zum dynamischen Füllen der Lücken im DOT-Template genutzt werden. Dadurch entsteht ein individuell angepasster DOT-Graph, der die spezifischen Daten widerspiegelt.

Der dritte Schritt umfasst die Übergabe der gefüllten Daten an das Go-Template `artifacts/show.tmpl`. Die Übergabe der Daten erfolgt mithilfe des Web-Frameworks *Gin*. Dieses Go-Template ist für die Generierung des finalen HTML-Codes zuständig, der den DOT-Graph enthält [73].

Anschließend werden im vierten Schritt die Daten aus den übergebenen Variablen gelesen und in die für *Hotwire Stimulus* benötigten Felder eingefügt, um sie später in JS weiterverarbeiten zu können. Die Daten werden dabei als Werte in HTML-Attributen gesetzt und sind nicht direkt für den Nutzer sichtbar. *Stimulus* benötigt hierbei Daten im Attribut `data-xxx-value`. Dieses Attribut enthält den DOT-String. Außerdem wird in `data-xxx-target` der Name des HTML-Elements definiert, der als Zielcontainer für die Visualisierung des Graphen dient. Der JS-Controller von Hotwire sorgt dafür, dass die Daten dynamisch an das Zielelement übergeben werden, wodurch die Interaktivität der Darstellung gewährleistet wird [74].

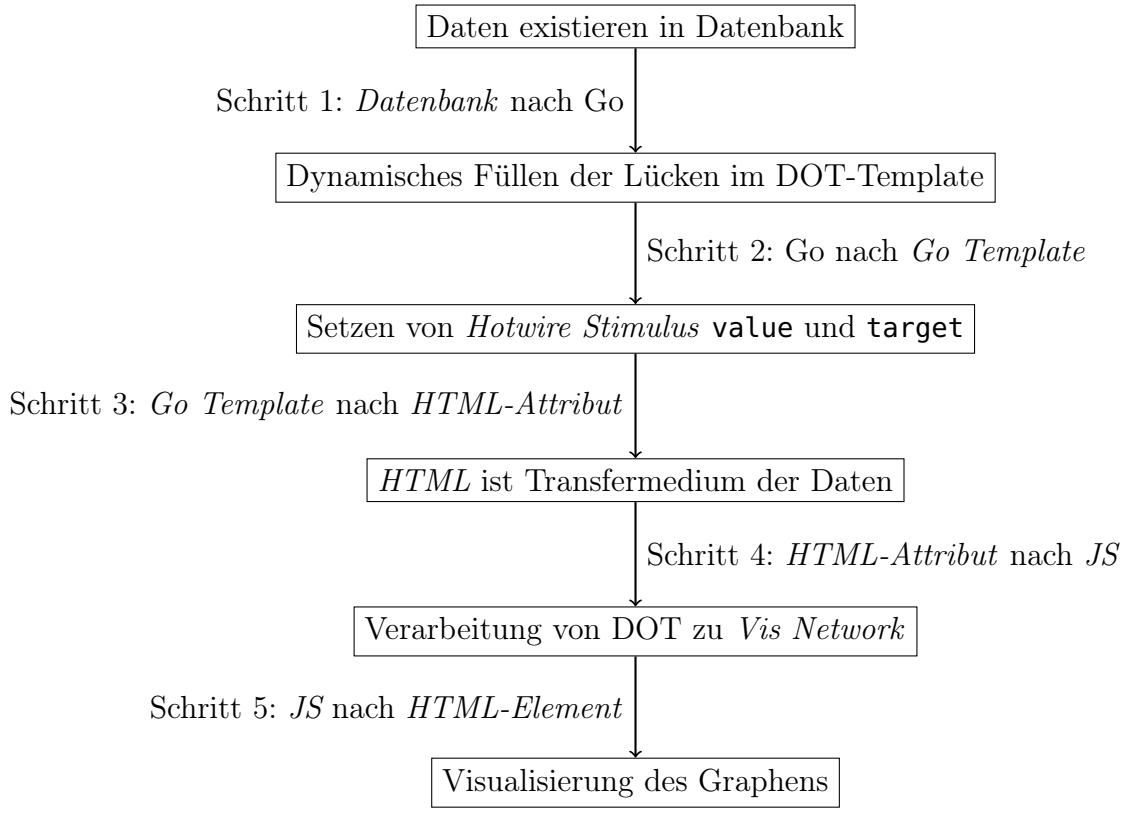

Abbildung 5.6.: Schritte zur Erstellung des Graphens

Im letzten Schritt übernimmt der JS-Controller, in diesem Fall vis_controller.js, die Verantwortung. Dieser liest den dynamisch generierten DOT-String aus dem HTML-Attribut aus und verarbeitet sie mit der Funktion `parseDOTNetwork` aus dem JS-Paket vis-network weiter. Dieses Paket wandelt den DOT-Quellcode in eine visuelle Graphendarstellung um, die anschließend in das Zielelement eingefügt wird. Durch die Verwendung von vis-network wird eine Darstellung gewährleistet, die es dem Nutzer erlaubt mit dem Graphen zu interagieren [75]. Detaillierte Informationen zu den Funktionen des Graphens wurden in Kapitel 4.2 gegeben.

Insgesamt bietet dieser Prozess eine effiziente Methode, um DOT-Quellcode dynamisch zu erstellen, in JS zu interaktiven Graphen zu verarbeiten und in HTML visuell darzustellen.

5.3. Sonstige Implementierungen

Neben den bereits genannten Themen wurden viele weitere kleinere Änderungen am Dashboard vorgenommen, um die Funktionalität und Benutzererfahrung zu verbessern. Auf eine Auswahl dieser sonstigen Verbesserungen wird in diesem Kapitel eingegangen.

5.3.1. Weiterführende Informationen zu Techniken

Wie in Kapitel 3.6 beschrieben, bieten sich drei wissenschaftliche Konstruktionen an, aus denen Techniken in die Matrix im Dashboard übernommen werden können. Bis auf zwei Techniken können alle im Dashboard angezeigten Techniken mindestens zwei Techniken aus TM4K, MITRE ATT&CK Matrix oder dem WG11 Report zugeordnet werden. Wie in 5.1.1 beschrieben, wurde diese Zuweisung über einen Vergleich der Beschreibungen der jeweiligen Techniken manuell vorgenommen. Um diese Zuweisungen nicht nur ohne weitere Funktion textuell darzustellen, wurde eine Verlinkung zu weiterführenden Informationen auf den Seiten [7] und [30] hinzugefügt. Eine Onlinequelle zu den O-RAN Bedrohungen gibt es bisher nicht, diese sind alleine in einem Microsoft Word Dokument veröffentlicht. Die URL, an die nach einem Klick weitergeleitet wird, ist dynamisch aus den Daten der Technik zusammengesetzt. Für die MITRE ATT&CK Webseite setzt sich die URL so zusammen:

```
attack.mitre.org/techniques/{Technik-ID}/{Subtechnik-ID}
```

Microsofts TM4K Seite nutzt für die URLs nicht die eindeutige ID, sondern den Titel der Technik. Die URL kann also wie folgt definiert werden:

```
microsoft.github.io/[...]/techniques/{Technik-Titel}
```

Die Zuordnung zu den Quellen wird auch farblich gekennzeichnet. Wenn kein Wert angezeigt wird, ist keine weitere Zuordnung möglich.

5.3.2. Verbesserung der Zeitleiste

Die Zeitleiste ist ein wichtiges Feature im Dashboard. Sie zeigt visuell aufbereitet, in welchem Zeitraum ein Angriff ausgeführt wurde. Die Implementierung bringt jedoch einige technische Herausforderungen mit sich, für die im Folgenden eine Lösung präsentiert wird.

```
1    options.Find().SetSort(bson.D{{Key: "timestamp_start", Value: -1}})
```

Listing 5.6: Sortierung der Datenbankobjekte

Ein Angriff dauert in der Regel nur wenige Sekunden, eine Darstellung auf einer Zeitleiste, die mehrere Stunden oder Tage umfasst, ist daher schwierig. Das JS Framework *Apexcharts.js*, welches für die Visualisierung der Zeitleisten verwendet wird, bietet eine Funktion die es erlaubt, auch kleinste zeitliche Abschnitte in einem großen Zeitraum zu erkennen. Die Funktion fügt zum Start- und Endzeitpunkt eines Angriffs jeweils einen Punkt hinzu, dessen Größe nicht mit der Dauer des Angriffs skaliert. In einer Übersicht über mehrere Stunden oder Tage ist jedes Artefakt mit einem gleichgroßen Punkt gekennzeichnet. Eine detaillierte Ansicht kann dann über die Zoomfunktion erreicht werden. Dabei kann direkt in der Zeitliste durch *Klicken-und-Ziehen* der Maus eine Zeitspanne ausgewählt werden. Diese beiden Funktionen werden über die Konfiguration in der Datei `assets\js\controllers\timeline_controller.js` aktiviert.

Bei der Darstellung einer großen Menge von Artefakten auf der Zeitleiste ist eine deutliche Einbuße der Performance zu spüren. Über subjektives Empfinden wurde eine Grenze von 100 Artefakten gefunden, bis zu der die Visualisierung ohne merkliche Verzögerung möglich ist. Die Begrenzung wird im Logik Teil der Anwendung angewendet, bevor die Daten überhaupt vom Server in die HTML Datei eingebettet werden. Die Datenbankabfrage sortiert die Daten absteigend nach Startdatum, es werden daher die 100 Artefakte angezeigt, welche die Daten der zuletzt ausgeführten Angriffe enthalten [76]. Der dazugehörige Quellcode ist in Listing 5.6 dargestellt.

5.3.3. Vereinfachung der Datenstruktur eines Artefakts

Über den Zeitraum der Implementierung haben sich nicht wenige Änderungen in der Datenstruktur eines Artefakts ergeben. In einer NoSQL-Datenbank, einem nicht-relationalen Datenbanksystem, das oft flexiblere Datenmodelle erlaubt, muss prinzipiell keine feste Struktur vorhanden sein, diese ist jedoch für das Auslesen eines Objekts in die definierte Datenstruktur in Go nötig. Da die Serialisierung von Python zu JSON in einer anderen Anwendung und von anderen Personen implementiert wurde als die Deserialisierung von JSON zu Go, war eine Absprache von großer Wichtigkeit. So wurde aus dem Feld `mitre` mit Datentyp `String` zuerst eine Liste um auch eine Zuordnung zu mehreren MITRE Techniken abbilden zu können. Später wurden alle Informationen über Techniken und Taktik in ein Feld vereint. Das Feld `command` wurde angepasst, um die Ausführung mehrerer Kommandos

```
type Artifact struct {
        ID          primitive.ObjectID  `bson:"_id" json:"id"`
        IP          string              `bson:"ip" json:"ip"`
        ...
        NewField    float64             `bson:"new_field" json:"new_field"`
    }
```

Listing 5.7: Hinzufügen eines Feldes zum Schema in Go

während eines Angriffs modellieren zu können. Weitere kleinere Änderungen führten zu dem Schema, welches im Anhang F dargestellt ist. Eine Erweiterung dieses Schemas ist einfach möglich. Das hinzugekommene Feld muss dafür in der Datei `pkg\artifacts\model.go` in der `struct` Definition eines Artefakts hinzugefügt werden, wie in Listing 5.7 demonstriert ist.

5.3.4. Verknüpfung zwischen Kampagnen und Artefakten

Kampagnen sind ein Bestandteil des ATs die es ermöglichen, mehrere Angriffe zu einer übergeordneten Einheit zusammenzufassen. In der Datenbank sind Kampagnen und Artefakte über eine *One-to-Many* Relation implementiert. Bisher war es im Dashboard nicht möglich die Artefakte aufzulisten, die einer Kampagne zugeordnet sind. Über eine einzelne Datenbankabfrage können die Artefakte gefunden werden, die einer Kampagne zugeordnet sind. Diese werden jetzt auf der Detailseite einer Kampagne angezeigt. Außerdem wurde eine Zeitleiste hinzugefügt, die diese Artefakte im Zeitraum der Laufzeit der Kampagne anzeigt. Dafür wurde die bestehende Logik der Artefaktzeitleiste auf der Übersichtseite aller Artefakte wiederverwendet. Das Ergebnis ist in Abbildung 5.7 dargestellt. Diese Funktion setzt die übergreifende Anforderung um, alle relevanten Daten auch über das Dashboard abrufen zu können, ohne auf die Datenbank direkt mithilfe der *MongoDB Query Language* oder anderer externer Tools zugreifen zu müssen.

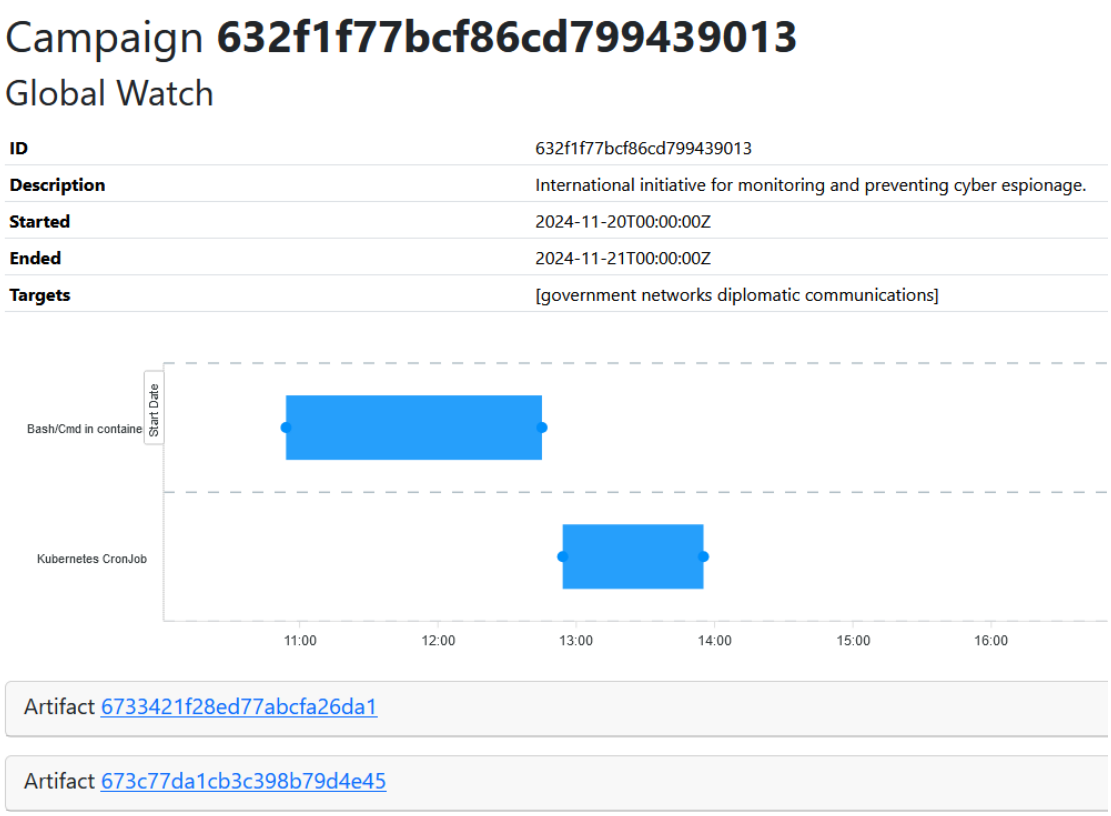

Abbildung 5.7.: Zuordnung von Artefakten zu einer Kampagne und Visualisierung der Artefakte auf einer Zeitleiste.

6. Diskussion

6.1. Interpretation der Ergebnisse

In diesem Kapitel werden die von ACEMA generierten Diagramme interpretiert sowie die Ergebnisse der Implementierung des Dashboards bewertet.

6.1.1. Interpretation der ACEMA Diagramme

Über das Analyseskript der ACEMA Implementierung lassen sich verschiedene Arten von Diagrammen erstellen. Für die Analyse der Daten wurden Diagramme der Typen Heatmap-, Netz- und Balkendiagramm erstellt. Eine Auswahl von besonders relevanten Diagrammen ist in Anhang H zu finden. Alle Diagramme sind über das Github-Repository [77] abrufbar und über den bereitgestellen Quellcode reproduzierbar. Im Folgenden werden drei der Diagramme genauer betrachtet und interpretiert.

Das Netzdiagramm in Abbildung H.1 zeigt den durchschnittlichen CVSS Vektor der Metriken aus der Basisgruppe gruppiert nach dem Schweregrad (*baseSeverity*). Dieser wird aus dem arithmetischen Mittel aller Vektoren ($N = 353$) berechnet, die mittels ACEMA gefunden werden. Die Metriken *Access Vector*, *Access Complexity* und *Authentication* geben an, wie auf die Schwachstelle zugegriffen wurde und ob für den Zugriff weitere Hürden überwunden werden mussten. Die *Impact*-Metriken geben an, wie schwer die Auswirkung in den jeweiligen Kategorien nach erfolgreicher Ausnutzung wären [65]. Zum Verständnis des Netzdiagramms ist die Erklärung in Anhang H über die Umwandlung von textueller zu numerischer Darstellung hilfreich. Vereinfacht ergibt sich die Beobachtung: Je weiter das Netz nach *außen* spannt, desto *gefährlicher* ist ein CVE.

Der Zugriff auf die durchschnittliche Schwachstelle bei mittlerem ($n = 154$) oder hohem ($n = 191$) Schweregrad kann in den meisten Fällen über das Netzwerk erfolgen. Eine weitere Authentifizierung muss nicht umgangen werden, um Zugriff zu erlangen. Die Komplexität des Zugriffs befindet sich zwischen hoher und mittlerer Komplexität. Die Auswirkungen sind bei hohem Schweregrad jedoch um ungefähr 0,25 Punkt größer als bei mittlerem Schweregrad, und beeinträchtigen die Integrität,

Verfügbarkeit und Vertraulichkeit des Systems durchschnittlich teilweise. Bei CVEs mit niedrigem Schweregrad ist eine deutliche Zunahme der Zugriffskomplexität erkennbar. Dies hängt auch damit zusammen, dass der initiale Zugriff häufiger nicht über das direkte Netzwerk möglich ist, sondern über benachbarte Netzwerke erfolgt. Mit Ausnahme der Auswirkung auf Integrität sind die Auswirkungen nochmals deutlich schwächer als bei mittlerem Schweregrad. Der Ausreißer könnte auf die geringe Stichprobe von $n = 8$ aus der Grundgesamtheit von $N = 353$ zurückzuführen sein.

Es ist aus der Anzahl der Stichproben pro Schweregrad deutlich erkennbar, dass den im Dashboard dargestellten MITRE-Techniken signifikant häufiger CVEs mit hohem oder mittlerem Schweregrad zugeordnet werden als solche mit niedrigem Schweregrad. Das lässt sich mit der Tatsache verknüpfen, dass Open RAN Systeme standardmäßig mit dem Netzwerk verbunden sind, um ihrer Funktion als Netzwerkinfrastruktur gerecht zu werden. Diese Schnittstelle erhöht die Möglichkeit und reduziert die Komplexität eines Zugriffs.

Abbildung H.2 zeigt den Durchschnitt der CVSS Gesamtbewertung pro Schweregrad und MITRE-Technik. Auffallend sind hier vier Techniken, bei denen der Durchschnitt der Bewertung bei hohem Schweregrad 8 übersteigt. Zwei der vier Techniken werden im Folgenden genauer betrachtet.

Eine der Techniken ist *T1059*, die in der TM4K die Bezeichnung *Bash or cmd inside container* trägt [78]. Es handelt sich dabei um Angriffe, die es erlauben, bösartigen Code innerhalb eines Containers in einer Kubernetes Umgebung aufzuführen. In Zusammenhang mit Zugriff über das Netzwerk werden solche Angriffe auch als *Remote Code Execution* bezeichnet und haben meist schwere Auswirkungen auf das betroffene System.

Eine weitere Technik ist *T1528: Container service account*. Diese Technik befasst sich mit der Ausnutzung von Service-Konten, die zum Beispiel in Kubernetes für jede Einheit zur Ausführung von Containern angelegt werden. Service-Konten haben oft hohe Privilegien und ein Angriff über diese Technik kann verschiedene Taktiken verfolgen. Laut der TM4K kann diese Technik für den Zugriff auf andere Zugangsberechtigungen, laterale Bewegung innerhalb des Systems und Persistenz genutzt werden [79]. Diese beiden Techniken bergen hohes Potenzial für schwere Auswirkungen in Kubernetes Umgebungen.

Die Auswirkung auf spezielle Komponenten in O-RAN wird über die Abbildung H.3 in Verbindung mit dem Report der WG11 ersichtlich [12]. Die Abbildung zeigt die Summe der CVSS Gesamtbewertung über alle CVEs pro O-RAN *Threat-ID* und Schweregrad. Auffallend sind vor allem die Summierungen, die einen Wert über 500 erreichen. Genaue Werte sind in Tabelle 6.1 dargestellt. Die höchste Summe

erreicht dabei *T-VM-C-01*, die in [12] den Titel *Abuse of a privileged VM/Container* trägt. Die *Threat-ID* wurde in der manuellen Zuordnung acht MITRE-Techniken zugeordnet, darunter auch *T1059* (vgl. Abbildung H.2). Über den Report der O-RAN WG11 können Komponenten einer virtualisierten Umgebung und spezielle Komponenten aus der O-RAN Architektur (siehe Abbildung 3.1) identifiziert werden, die von einem Angriff aus dieser Kategorie betroffen sind. Für die genannte *Threat-ID* sind das der *Near-Real-Time* RAN Intelligent Controller (RIC), O-RAN Central Unit (O-RU) und O-RAN Radio Unit (O-DU) sowie das Hostbetriebsystem und die Virtualisierungsplattform.

Tabelle 6.1.: Top 5 der O-RAN Threat-IDs, dessen zugeordnete CVEs eine hohe Summierung ihrer Werte nach CVSS zeigen.

Score	O-RAN Threat
645.7	T-VM-C-01
570.9	T-IMG-01
560.4	T-ADMIN-02
388.7	T-O2-01
348.6	T-HW-01

6.1.2. Erfolg der Implementierung

Die Implementierung wurde ohne große Hindernisse erfolgreich beendet. Alle Meilensteine wurden wie geplant erreicht. Besonders die Integration der Methode von Klement, Liu und Katzenbeisser brachte einen großen Mehrwert für das Dashboard. Die Visualisierung der CVSS Daten in der Matrix und in der Detailseite von Artefakten liefen wertvolle Einblicke in die Bewertung der Auswirkungen der Angriffe. Die erstellten und im Dashboard angezeigten Diagramme liefern sowohl technische als auch wissenschaftliche Erkenntnisse.

Ein zentraler Fortschritt bei der Weiterentwicklung des Dashboards war die Visualisierung des Angriffspfads, der als interaktiver Graph die kritischen Informationen zu Angriffsmustern, Schwachstellen und genutzten Werkzeugen kompakt darstellt. Neben den *MUST-Features* wurden weitere Funktionen wie die Verbesserung der Zeitleistenansicht und die Verlinkung zu weiterführenden Information über Techniken implementiert, welche die Benutzbarkeit verbessern und den Funktionsumfang des Dashboards erweitern. Insgesamt zeigt das Ergebnis, wie eine wissenschaftliche Herangehensweise an die Konzeption und Implementierung eines Dashboards die Analyse- und Visualisierungsfähigkeiten in einem DFIR-Umfeld erheblich verbes-

sern kann. Die erfolgreiche Umsetzung belegt die Praxistauglichkeit der theoretisch erarbeiteten Methoden.

6.1.3. Einsatz in der Praxis

In Kapitel 4.1 wird die Existenz von anderen Mappingtools erwähnt. Diese erfüllen teilweise dieselbe Funktion wie ACEMA, bieten aber keine Möglichkeit der Integration von Bedrohungsdaten aus der spezialisierten O-RAN Umgebung.

Die umfangreichste Ansammlung von Zuordnungen wird im *Mappings Explorer* vom MITRE *Center for Threat-Informed Defense* gepflegt. Dieses Tool bietet nicht nur eine Möglichkeit zur Zuordnung von MITRE-Techniken zu CVEs und umgekehrt, sondern auch in weiterer Frameworks, die eine Art der Kategorisierung von Schwachstellen implementieren. Solche eigenen Frameworks werden zum Beispiel von großen Anbietern für Clouddienste wie Google, Amazon und Microsoft konzipiert. Mappings in diese Frameworks sind über den *Mappings Explorer* erstellt und abrufbar [80]. Eine Integration mit dem O-RAN Bedrohungsmodel gibt es bisher nicht [81].

Ein simpleres Tool wird vom Forschungsteam *Voyager18* des Unternehmens *Vulcan Cyber* bereitgestellt. Es erlaubt das Mapping zwischen MITRE-Technik und CVEs, und umgekehrt [82]. Laut [83] verbindet das Tool maschinelles Lernen und textuelle Analyse zum Finden von Mappings und soll für weitere Anwendungszwecke weiterentwickelt werden.

Eine stichprobenartige Analyse der gespeicherten Mappings der drei Mappingtools fand keine Übereinstimmung zwischen identifizierten CVEs für MITRE-Techniken. Diese Abweichungen sind auf die Anzahl von über 240.000 CVEs zurückzuführen. Durch die textuelle Analyse oder maschinelles Lernen ist es möglich, dass die Tools unterschiedliche CVEs für eine Technik oder umgekehrt finden. Die Unterschiede könnten auch aufgrund der Verwendung von verschiedenen Datenquellen entstehen. ACEMA findet eine ausreichende Datenmenge, um eine aussagekräftige Datenanalyse durchzuführen, würde aber durch einen größeren Datensatz weiter verbessert werden. Die Implementierung des Dashboards ist aktuell schon in der Lage, in einer realen Umgebung eingesetzt zu werden, um die effektive Bewertung von Angriffen zu ermöglichen.

6.2. Limitation

Die Analyse und Implementierung unterliegt spezifischen Einschränkungen, die sowohl durch methodische als auch durch technologische Faktoren bedingt sind. Dabei sind insbesondere die Qualität und die Aktualität der genutzten Datenquellen zu betrachten.

6.2.1. Generelle Limitationen

Die NVD gilt als zentrale Ressource für Informationen über öffentlich bekannte Sicherheitslücken. Daher scheint es auch als verlässliche Quelle für die IT-Sicherheitsbranche und Forschungszwecke. Laut einer Veröffentlichung vom 13. Februar 2024 kommt das NIST nicht der Abarbeitung der eingereichten CVEs hinterher und arbeitet daran, die Verzögerung bei der Analyse zu beheben [84]. Am 13. November 2024 kündigte das NIST an, alle neu eingereichten CVE direkt verarbeiten zu können. Die Abarbeitung der zurückgestellten Daten dauert weiterhin an. Die Daten aus der NVD sind daher aktuell nicht auf dem neusten Stand [85]. Eine vom *IT Security Infrastructures Lab* der *Friedrich-Alexander Universität Erlangen-Nürnberg* veröffentliche Studie vom 26. Okt. 2024 unterlegt diese Feststellung mit empirischen Daten. Demnach sind fast 40% der Befragten jährlich, monatlich, wöchentlich oder täglich von der verzögerten Veröffentlichung von neuen CVEs betroffen [86]. Die damit zusammenhängenden und weitere technische Schwierigkeiten beeinflussen auch die Abfrageverfügbarkeit und Abfragegeschwindigkeit von Daten aus der NVD. Für ACEMA haben die Einschränkungen weitreichende Konsequenzen, von verlangsamter Abfragegeschwindigkeit des CVSSs für CVEs bis hin zur vollständigen Abwesenheit von CVSS Daten, die eine integrale Rolle für die Datenanalyse spielen.

In der Sicherheitsforschung ist es essenziell, stets den neuesten Stand der Angriffe zu verfolgen, da Bedrohungslandschaften und Technologien sich kontinuierlich weiterentwickeln. Die Aktualität der verwendeten Methoden und Tests hat dabei unmittelbare Auswirkungen auf die Aussagekraft der Forschungsergebnisse. Insbesondere im Bereich Kubernetes nutzt das AT die Kommandos, die von der Redguard KTM zur Verfügung gestellt werden. Allerdings zeigt der Inhalt des HTTP-Headers `Last-Modified`, dass die genutzte Quelle seit etwa zwei Jahren nicht aktualisiert wurde, wie in Listing 6.1 zu sehen ist.

Die Änderungen, die am 7. Dezember 2022 in der Microsoft TM4K vorgenommen wurden, wurden nie in die KTM übernommen und es sind keine Kommandos vorhanden um die spezifischen Angriffe aus zwei neuen Techniken (*Static Pods* und

```
$ curl -I -k -s https://kubernetes-threat-matrix.redguard.ch/ | grep -i "Last-Modified"
> last-modified: Fri, 04 Nov 2022 10:51:23 GMT
```

Listing 6.1: Abfrage des letzten Änderungsdatums der Kubernetes Threat Matrix von RedGuard

Collecting data from pod) zu testen [87]. Zwischen dem 1. Januar 2023 und dem 10. Dezember 2024 wurden auf der Kubernetes-eigenen Webseite 18 CVEs veröffentlich, auf die folglich nicht über die in der Redguard KTM verfügbaren Kommandos getestet werden kann [88]. Das Update vom 7. Dezember stellt auch die letzte dokumentierte Änderung an der TM4K dar [89]. Wenn alle in der Zwischenzeit entdeckten Schwachstellen zu vorhandenen Techniken der TM4K zugeordnet werden konnten, ist prinzipiell keine Aktualisierung nötig.

6.2.2. Limitation von ACEMA

Die Implementierung von ACEMA nutzt für die Bewertung das CVSS in Version 2 aus dem Jahr 2007 [90]. Seit dem 27. Juni 2024 unterstützt die NVD das Bereitstellen von Daten in *v4.0* des CVSSs [91]. Nicht nur ist die in ACEMA genutzte Version durch zwei *major*-Versionen überholt, seit dem 13. Juli 2022 werden in der Datenbank keine neuen Angriffsvektordaten, qualitative Schweregrad-Bewertungen oder Schweregrad-Werte in CVSS Version 2 erstellt [92]. Der Fall von unvollständigen Daten ist in der Bearbeitung dieses Projekts nicht vorgekommen, da alle im Ergebnisse des Mappings aufgeführten CVEs im Jahr 2021 oder davor veröffentlicht wurden. Es ist jedoch theoretisch möglich, dass diese Einschränkung bei der weiteren Nutzung dieser Methode zu fehlenden Daten führen kann.

Das von ACEMA, mithilfe des CTI Repositories, erstellte Mapping von MITRE-Technik zu CAPEC (Schritt 1 in Abbildung 4.1) findet nicht alle CAPECs. Die Dokumentation im Repository erklärt zu dem Finden von CAPECs: CAPEC IDs können dort für die Techniken gefunden werden, wo unter externen Referenzen das Attribut *source_name* den Wert `capec` hat [36] [93]. Klement, Liu und Katzenbeisser nutzen in ihrer Implementierung die Daten aus dem *enterprise-attack/attack-pattern* Teil des Repositories [54]. Nach demselben Prinzip kann auch der *capec/2.1/attack-pattern* Teil des Repositories durchsucht werden. Hauptsächlich liegt der Unterschied zwischen den beiden Datensätzen darin, was der Hauptzweck der Daten ist. Der *enterprise-attack* Teil enthält die Datenstruktur und Inhalte der MITRE ATT&CK *Enterprise*-Matrix, mit Daten wie der MITRE-Technik ID, Taktik und angreifbare Plattform sowie manchmal Referenzen zu MITRE-Techniken. Der CAPEC Teil enthält die Datenstruktur und Inhalte des CAPEC-Frameworks,

mit Daten wie CAPEC ID, Konsequenzen und Voraussetzungen für die Ausnutzung sowie manchmal Referenzen zu MITRE-Techniken. Es zeigt sich in den Daten aus Tabelle 6.2, dass über den *CAPEC* Teil eine vollständigere Abbildung von MITRE-Technik zu CAPEC ID möglich ist. Die beiden Implementierungen des Mappings wurden auf demselben Datensatz aus Anhang D ausgeführt, der ein direkter Export der Daten aus der Dashboardmatrix ist.

Tabelle 6.2.: Vergleich der Daten zwischen eigener Implementierung und ACEMA Implementierung

Metrik	**Eigene Impl.**	**ACEMA Impl.**	**Differenz (%)**
Techniken	27	27	0 %
Techniken mit CAPEC	14	3	-75 %
Einzigartige CAPECs	25	6	-76 %
Einzigartige CWEs	39	15	-62 %

6.3. Ausblick

Trotz des bereits hohen Funktionsumfangs und guter Anwendbarkeit des Dashboards gibt es Verbesserungspotenzial. Eine wesentliche Schwäche liegt in der manuellen Zuordnung bestimmter Daten wie der O-RAN *Threat-IDs*, was zeitaufwändig und fehleranfällig ist. Hier könnte eine stärkere Automatisierung durch Machine-Learning-Ansätze Abhilfe schaffen. Klement, Liu und Katzenbeisser kündigen in ihrem Artikel an, dass sich ihre zukünftigen Forschungen stark mit diesem Thema beschäftigen werden, mit dem Ziel den gesamten Mappingprozess zu automatisieren [9]. Der Einsatz von Daten aus CVSS in Version 2 sollte bei anhaltender Nutzung der Methode eingestellt und zugunsten von Version 4 ersetzt werden. Die Version 4 des CVSSs wurde am 1. November 2023 veröffentlicht und weiterhin fortlaufend verbessert. Die neue Version bietet laut Dugal eine feinere Granularität und viele weitere Verbesserungen [94,95]. Zukünftig wäre die Implementierung einer Live-Aktualisierung für Artefaktdaten und Angriffspfadvisualisierungen wünschenswert, um auch bei dynamischen Angriffssimulationen stets aktuelle Informationen bereitzustellen. Ein weiteres Verbesserungspotenzial besteht in der Performanceoptimierung, insbesondere bei der Darstellung großer Datenmengen, wie sie bei der Analyse umfangreicher Artefaktsammlungen auftreten können. Hier könnten zum Beispiel bewährte Methoden wie *Chunking* oder *Lazy Loading* eingesetzt werden, um große Datenmengen nacheinander in kleineren *Chunks* zu laden.

Die Funktionalität des Dashboards ist nicht an die Anwendung auf eine bestimmte Technologie gebunden, sondern kann auf weitere Umgebungen übertragen werden.

Die Daten aus ACEMA können weiter genutzt werden, solange sich die Angriffe mithilfe von MITRE-Techniken ausdrücken lassen. Es ist daher denkbar, das Dashboard in Verbindung mit dem etablierten AT-Framework in der sich verändernden Open RAN Umgebung auch zukünftigen mit Technologien wie 6G zu nutzen.

7. Zusammenfassung

In dieser Arbeit wurde ein Dashboard implementiert, das speziell für den Einsatz in einer DFIR Umgebung für Open RAN konzipiert wurde. Ziel war es, ein Werkzeug zu schaffen, das sowohl die Durchführung von Pentesting-Szenarien unterstützt als auch die Bewertung von Schwachstellen in Open RAN-Systemen ermöglicht. Die Weiterentwicklung basiert auf der Vorarbeit aus der vorherigen Forschung im 5G-FORAN-Projekt und der Integration der empirischen Methode ACEMA, die durch ihre Fähigkeit zur Analyse und Visualisierung von Schwachstellen wesentlich zur Funktionalität des Dashboards beiträgt. Die Ergebnisse belegen die technische und wissenschaftliche Relevanz des Dashboards für den Einsatz in einer DFIR Umgebung.

Die Implementierung des Dashboards wurde erfolgreich abgeschlossen und alle Meilensteine planmäßig erreicht. Zu den zentralen Ergebnissen zählt die Integration von ACEMA, welche durch die Nutzung von CVSS-Daten eine präzise Bewertung von Schwachstellen in Open RAN Umgebungen ermöglicht. Die dynamischen Visualisierungen unterstützen die Nutzer dabei, Beziehungen zwischen spezifischen Angriffen und Techniken aus etablierten Frameworks wie MITRE ATT&CK schnell zu erkennen.

Die Analyse der von ACEMA generierten Diagramme zeigte, dass Open RAN-Systeme durch standardmäßige Netzwerkanbindung ein erhöhtes Angriffspotenzial aufweisen. Beispielsweise zeigte die Analyse, dass hochkritische Angriffsmuster wie *Remote Code Execution* in Open RAN Umgebungen besonders relevant sind, da sie über das Netzwerk ausgenutzt werden können und schwerwiegende Auswirkungen auf die Integrität, Verfügbarkeit und Vertraulichkeit der Systeme haben. Die Ergebnisse verdeutlichen auch, dass die meisten relevanten Schwachstellen mit einem mittlerem oder hohem Schweregrad bewertet werden und spezifische Bedrohungen wie die Ausnutzung von Container-Service-Konten oder privilegierten VMs besonders gefährlich sind. Dies unterstreicht die Bedeutung einer regelmäßigen Bedrohungsanalyse und einer Implementierung nach dem *security-by-design*-Prinzip in Kubernetes-basierten Umgebungen.

Mit der Entwicklung dieses Dashboards wurde ein wichtiger Beitrag zur Sicherheits-

forschung im Bereich Open RAN geleistet, indem ein praktisches *Open-Source*[1]-Werkzeug für die Visualisierung und Analyse von Schwachstellen bereitstellt wird. Die Relevanz einer empirischen Analyse und Kategorisierung von Schwachstellen in unterschiedlichen technologischen Kontexten wird durch die Studien von [16] und [9] verdeutlicht. Die Ergebnisse dieser Arbeit ergänzen die Studie von Klement, Liu und Katzenbeisser und zeigt, wie wissenschaftliche Methoden in praxisnahe Anwendungen überführt werden können.

Mit dem Dashboard wurde ein effektives Werkzeug für die Sicherheitsanalyse in Open RAN Umgebungen implementiert. Die gewonnenen Erkenntnisse bieten nicht nur praktische Ansätze zur Verbesserung der Sicherheit von Open RAN, sondern dienen auch als Grundlage für zukünftige Forschung und Entwicklung in DFIR Umgebungen.

[1] Siehe Anhang A.

Literaturverzeichnis

[1] *O-RAN Downloads*. Spezifikationen der O-RAN Alliance. URL: https://specifications.o-ran.org/specifications (besucht am 12. 11. 2024).

[2] Mi-jin Kim, Doyoung Eom und Heejin Lee. „The Geopolitics of next Generation Mobile Communication Standardization: The Case of Open RAN". In: *Telecommunications Policy* 47.10 (1. Nov. 2023), S. 102625. ISSN: 0308-5961. DOI: 10.1016/j.telpol.2023.102625. URL: https://www.sciencedirect.com/science/article/pii/S0308596123001362 (besucht am 10. 12. 2024).

[3] O-RAN Alliance. *O-RAN.White-Paper-2018-10*. 2018. URL: https://mediastorage.o-ran.org/white-papers/O-RAN.White-Paper-2018-10.pdf (besucht am 10. 12. 2024).

[4] *5G-FORAN*. Homepage des 5G-FORAN Projekts. URL: https://www.5g-foran.com/mehr.html (besucht am 12. 11. 2024).

[5] Henrik Wittemeier, Thomas Karl, Arn Jonas Dieterich und Andreas Grebe. „Digital Forensics and Incident Response (DFIR) in O-RAN Implementations". In: *Mobilkommunikation; 28. ITG-Fachtagung*. Mobilkommunikation; 28. ITG-Fachtagung. Mai 2024, S. 25–30. URL: https://ieeexplore.ieee.org/abstract/document/10651556 (besucht am 09. 11. 2024).

[6] *Solving Problems for a Safer World | MITRE*. Homepage von MITRE. 31. Okt. 2024. URL: https://www.mitre.org/ (besucht am 13. 11. 2024).

[7] *MITRE ATT&CK®*. Homepage von MITRE ATT&CK. URL: https://attack.mitre.org/ (besucht am 12. 11. 2024).

[8] Jonas Dieterich. *Development of an Adversary Simulation Strategy for a Kubernetes-based Open RAN Deployment*. 18. Juni 2024. URL: https://www.dn.th-koeln.de/wp-content/uploads/2024/06/research_project_dieterich_11135427.pdf (besucht am 09. 11. 2024).

[9] Felix Klement, Wuhao Liu und Stefan Katzenbeisser. „Toward Securing the 6G Transition: A Comprehensive Empirical Method to Analyze Threats in O-RAN Environments". In: *IEEE Journal on Selected Areas in Communications* 42.2 (Feb. 2024), S. 420–431. ISSN: 0733-8716, 1558-0008. DOI: 10.1109/JSAC.2023.3339172. URL: https://ieeexplore.ieee.org/document/10339923/ (besucht am 23. 10. 2024).

[10] *About O-RAN Alliance*. Über-Seite der O-RAN Alliance. URL: https://www.o-ran.org/about (besucht am 12. 11. 2024).

[11] *CVE - Search Results*. Suchergebnisse nach Stichwortsuche. URL: https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=o-ran (besucht am 04. 12. 2024).

[12] O-RAN Work Group 11 (Security Work Group). *O-RAN Security Threat Modeling and Risk Assessment 4.0*. Okt. 2024. URL: https://specifications.o-ran.org/download?id=774 (besucht am 12. 11. 2024).

[13] Stefan Köpsell, Andrey Ruzhanskiy, Andreas Hecker, Dirk Stachorra und Norman Franchi. *Open-RAN Risikoanalyse*. BSI, 21. Feb. 2022. URL: https://www.bsi.bund.de/dok/5GRANR (besucht am 12. 11. 2024).

[14] European Union Agency for Network and Information Security. *ENISA Threat Landscape for 5G Networks: Threat Assessment for the Fifth Generation of Mobile Telecommunications Networks (5G)*. LU: Publications Office, 2019. URL: https://data.europa.eu/doi/10.2824/49299 (besucht am 15. 11. 2024).

[15] *EU Coordinated Risk Assessment of the Cybersecurity of 5G Networks*. 9. Okt. 2019. URL: https://ec.europa.eu/newsroom/dae/document.cfm?doc_id=62132 (besucht am 15. 11. 2024).

[16] Alejandro Mazuera-Rozo, Jairo Bautista-Mora, Mario Linares-Vásquez, Sandra Rueda und Gabriele Bavota. „The Android OS Stack and Its Vulnerabilities: An Empirical Study“. In: *Empirical Softw. Engg.* 24.4 (1. Aug. 2019), S. 2056–2101. ISSN: 1382-3256. DOI: 10.1007/s10664-019-09689-7. URL: https://doi.org/10.1007/s10664-019-09689-7 (besucht am 01. 12. 2024).

[17] O-RAN Work Group 1 (Use Cases and Overall Architecture). *O-RAN Architecture Description*. O-RAN Alliance, 29. März 2024. URL: https://specifications.o-ran.org/download?id=641.

[18] *K Release - Releases - Confluence/Wiki*. Dokumentation über das K-Release. URL: https://lf-o-ran-sc.atlassian.net/wiki/spaces/REL/pages/12812923/K+Release (besucht am 05. 12. 2024).

[19] Felix Klement, Alessandro Brighente, Michele Polese, Mauro Conti und Stefan Katzenbeisser. *Securing the Open RAN Infrastructure: Exploring Vulnerabilities in Kubernetes Deployments*. 3. Mai 2024. DOI: 10.48550/arXiv.2405.01888. arXiv: 2405.01888. URL: http://arxiv.org/abs/2405.01888 (besucht am 09. 11. 2024). Vorveröffentlichung.

[20] Felix Klement, Vincent Ulitzsch, Juliane Krämer, S. Stanczak, Zoran Utkovski, Igor Bjelakovic und Gerhard Wunder. „Open or Not Open: Are Conventional Radio Access Networks More Secure and Trustworthy than Open-RAN?“ In: (26. Apr. 2022). DOI: 10.48550/arXiv.2204.12227.

[21] Khalid Ali und Manar Jammal. „ML-Based Dynamic Scaling and Traffic Forecasting for 5G O-RAN“. In: *2023 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*. 2023 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech). Abu Dhabi, United Arab Emirates: IEEE, 14. Nov. 2023, S. 0444–0451. ISBN: 979-8-3503-0460-2. DOI: 10.1109/DASC/PiCom/CBDCom/Cy59711.2023.10361376. URL: https://ieeexplore.ieee.org/document/10361376/ (besucht am 04. 12. 2024).

[22] Johannes Franz Müller. *Implementation of a Framework for Generating Attack Traces in Open RAN Systems*. 15. Jan. 2024. URL: https://www.dn.th-koeln.de/wp-content/uploads/2024/06/Bachelorarbeit_johannes_franz_mueller.pdf.

[23] *MongoDB: The Developer Data Platform | MongoDB*. Homepage der MongoDB Software. URL: https://www.mongodb.com/ (besucht am 16. 11. 2024).

[24] *HTML Over The Wire | Hotwire*. Homepage des Hotwire Frameworks. URL: https://hotwired.dev/ (besucht am 06. 12. 2024).

[25] Jonas Levin Weber. *Evaluation of Dashboard Techniques for Digital Forensics and Incident Response (DFIR) in Open RAN*. URL: https://www.dn.th-koeln.de/wp-content/uploads/2024/06/research_project_report.pdf.

[26] *ATLAS Matrix | MITRE ATLAS™*. Homepage von MITRE ATLAS. URL: https://atlas.mitre.org/matrices/ATLAS (besucht am 07. 12. 2024).

[27] *CWE Website*. Homepage von MITRE CWE. URL: https://cwe.mitre.org/about/index.html (besucht am 02. 12. 2024).

[28] *CAPEC Website*. Homepage von MITRE CAPEC. URL: https://capec.mitre.org/about/index.html (besucht am 02. 12. 2024).

[29] *CVE Website*. Homepage von MITRE CVE. URL: https://www.cve.org/About/Overview (besucht am 02. 12. 2024).

[30] *Tactics - Threat Matrix for Kubernetes*. Homepage der Microsoft Threat Matix für Kubernetes. URL: https://microsoft.github.io/Threat-Matrix-for-Kubernetes/ (besucht am 19. 11. 2024).

[31] *Deployments · Microsoft/Threat-Matrix-for-Kubernetes*. GitHub. URL: https://github.com/microsoft/Threat-Matrix-for-Kubernetes (besucht am 25. 11. 2024).

[32] *Kubernetes Threat Matrix*. URL: https://kubernetes-threat-matrix.redguard.ch/ (besucht am 25. 11. 2024).

[33] *Assets - NVD Integration | Atlassian Support | Atlassian Documentation*. URL: https://confluence.atlassian.com/assetapps/asset-integration-nvd-1168848213.html (besucht am 25. 11. 2024).

[34] *Informationen Zu Den NVD-Integrationen*. URL: https://www.servicenow.com/docs/de-DE/bundle/vancouver-security-management/page/product/secops-integration-vr/nvd/concept/nvd-vuln-integration.html (besucht am 25. 11. 2024).

[35] *Mitre/Cti: Cyber Threat Intelligence Repository Expressed in STIX 2.0*. URL: https://github.com/mitre/cti (besucht am 19. 11. 2024).

[36] *Cti/USAGE.Md at Master · Mitre/Cti*. Anleitung zur Verwendung der CAPEC-Daten. URL: https://github.com/mitre/cti/blob/master/USAGE.md (besucht am 24. 11. 2024).

[37] *Aboutcode-Org/Cwe2*. Dokumentation über das Python Paket cwe2. 29. Aug. 2024. URL: https://github.com/aboutcode-org/cwe2 (besucht am 22. 11. 2024).

[38] *CWE - Downloads*. Downloadseite der CWE Liste. URL: https://cwe.mitre.org/data/downloads.html (besucht am 22. 11. 2024).

[39] *NVDLib :: NVDLib: NIST National Vulnerability Database API Wrapper*. Dokumentation über NVDLib. URL: https://nvdlib.com/en/latest/ (besucht am 19. 11. 2024).

[40] *MachineThing/Cve_lookup: Look up CVEs and Get Details about Them*. Dkumenation über das Python Paket cve_lookup. URL: https://github.com/MachineThing/cve_lookup (besucht am 19. 11. 2024).

[41] *NVD - Home*. Homepage der NVD. URL: https://nvd.nist.gov/ (besucht am 19. 11. 2024).

[42] Luisa Barrera-Leon, Fulvio Corno und Luigi De Russis. „How the Preattentive Process Is Exploited in Practical Information Visualization Design: A Review“. In: *International Journal of Human–Computer Interaction* 39.4 (25. Feb. 2023), S. 707–720. ISSN: 1044-7318. DOI: 10.1080/10447318.2022.2049137. URL: https://doi.org/10.1080/10447318.2022.2049137 (besucht am 16. 11. 2024).

[43] *Präattentive Wahrnehmung*. Definition des Begriffs. URL: https://biologie-seite.de/Biologie/Pr%C3%A4attentive_Wahrnehmung (besucht am 16. 11. 2024).

[44] Hanspeter A. Mallot. „Wahrnehmung, präattentive im Dorsch Lexikon der Psychologie“. In: (2021). URL: https://dorsch.hogrefe.com/stichwort/wahrnehmung-praeattentive (besucht am 16. 11. 2024).

[45] Alexandra Clifford, Amanda Holmes, Ian R.L. Davies und Anna Franklin. „Color Categories Affect Pre-Attentive Color Perception“. In: *Biological Psychology* 85.2 (Okt. 2010), S. 275–282. ISSN: 03010511. DOI: 10.1016/j.biopsycho.2010.07.014. URL: https://linkinghub.elsevier.com/retrieve/pii/S0301051110002139 (besucht am 16. 11. 2024).

[46] Edward R. Tufte und O. E. Katter. „Book Reviews: The Visual Display of Quantitative Information“. In: *IEEE Transactions on Professional Communication* PC-27.2 (Juni 1984), S. 108–108. ISSN: 0361-1434, 1558-1500. DOI: 10.1109/TPC.1984.6448807. URL: https://ieeexplore.ieee.org/document/6448807/ (besucht am 16. 11. 2024).

[47] Peyman Toreini und Moritz Langner. „DESIGNING USER-ADAPTIVE INFORMATION DASHBOARDS: CONSIDERING LIMITED ATTENTION AND WORKING MEMORY“. In: ().

[48] *Tooltip – Carbon Design System*. Beschreibung des Nutzens von Tooltips. URL: https://carbondesignsystem.com/components/tooltip/usage/ (besucht am 17. 11. 2024).

[49] *Vite*. Homepage der Vite Software. URL: https://vite.dev (besucht am 16. 11. 2024).

[50] *Add Example Data (5665bdad) · Commits · FORAN / Foran-Dash · GitLab*. GitLab. 22. Nov. 2023. URL: https://git.dn.fh-koeln.de/foran/foran-dash/-/commit/5665bdad90dacc48cb82ed95df4f97bb8d530103 (besucht am 16. 11. 2024).

[51] *JSON To MongoDB*. Dokumentation über das Laden von JSON nach MongoDB. URL: https://www.mongodb.com/resources/languages/json-to-mongodb (besucht am 12. 12. 2024).

[52] *Homepage | Ansible Collaborative*. Homepage der Ansible Software. URL: https://www.ansible.com/ (besucht am 16. 11. 2024).

[53] Ansible Collaborators. *How Ansible Works*. Beschreibung über die Funktionalität von Ansible. 8. Okt. 2024. URL: https://www.ansible.com/how-ansible-works/ (besucht am 16. 11. 2024).

[54] Stefan Katzenbeisser Felix Klement Wuhao Liu. „Towards Securing the 6G Transition: A Comprehensive Empirical Method to Analyze Threats in o-RAN Environments“. In: *2023 IEEE JSAC special issue on Open RAN* (2023). URL: https://github.com/fklement/acema_oran.

[55] *Welcome to Python.Org*. Homepage der Python Software. URL: https://www.python.org/ (besucht am 16. 11. 2024).

[56] *Project Jupyter*. Homepage der Jupyther Software. URL: https://jupyter.org (besucht am 29. 11. 2024).

[57] *Pandas - Python Data Analysis Library*. Homepage der Pandas Software. URL: https://pandas.pydata.org/ (besucht am 29. 11. 2024).

[58] *Mitreattack-Python/Mitreattack/attackToExcel/README.Md at Master · Mitre-Attack/Mitreattack-Python*. GitHub. URL: https://github.com/mitre-attack/mitreattack-python/blob/master/mitreattack/attackToExcel/README.md (besucht am 29. 11. 2024).

[59] *Introduction to STIX*. Dokumentation über das STIX Format. URL: https://oasis-open.github.io/cti-documentation/stix/intro (besucht am 19. 11. 2024).

[60] *Oasis-Open/Cti-Python-Stix2: OASIS TC Open Repository: Python APIs for STIX 2*. URL: https://github.com/oasis-open/cti-python-stix2 (besucht am 29. 11. 2024).

[61] *Developers - Start Here*. Dokumentation über die NVD API. URL: https://nvd.nist.gov/developers/start-here (besucht am 11. 12. 2024).

[62] *CAPEC - News & Events*. Neuigkeiten und Ankündigungen über CAPEC. URL: https://capec.mitre.org/news/index.html (besucht am 26. 11. 2024).

[63] *CAPEC CWE Mapping Schätzung*. 27. Nov. 2024. URL: https://chatgpt.com/share/674661e1-e2b8-8000-834c-aaa1eca10e60.

[64] *AI-Working-Group/Meeting_slides/20241115_CWE-AI-WG.Pdf at Main · CWE-CAPEC/AI-Working-Group*. 15. Nov. 2024. URL: https://github.com/CWE-CAPEC/AI-Working-Group/blob/main/meeting_slides/20241115_CWE-AI-WG.pdf (besucht am 27. 11. 2024).

[65] *CVSS v2 Complete Documentation*. FIRST — Forum of Incident Response und Security Teams. URL: https://www.first.org/cvss/v2/guide (besucht am 28. 11. 2024).

[66] *Embed Package - Embed - Go Packages*. URL: https://pkg.go.dev/embed (besucht am 01. 12. 2024).

[67] *Template Package - Text/Template - Go Packages*. Dokumentation über das template Paket. URL: https://pkg.go.dev/text/template (besucht am 29. 11. 2024).

[68] Contributors, Mark Otto, Jacob Thornton, and Bootstrap. *Badges*. Dokumentation über die Bootstrap Komponente Badges. URL: https://getbootstrap.com/docs/4.0/components/badge/ (besucht am 29. 11. 2024).

[69] *HTML Span Tag*. Dokumentation über das HTML Tag span. URL: https://www.w3schools.com/tags/tag_span.asp (besucht am 29. 11. 2024).

[70] *Digraph Definition*. Definition des Begriffs Digraph. URL: https://www.cs.odu.edu/~toida/nerzic/level-a/digraph/definition.html (besucht am 01. 12. 2024).

[71] *Graphviz*. Homepage der Graphviz Software. URL: https://graphviz.org/ (besucht am 01. 12. 2024).

[72] *DOT Language*. Homepage der Graphviz DOT Software. URL: https://graphviz.org/doc/info/lang.html (besucht am 01. 12. 2024).

[73] *Gin-Gonic/Gin: Gin Is a HTTP Web Framework Written in Go (Golang). It Features a Martini-like API with Much Better Performance – up to 40 Times Faster. If You Need Smashing Performance, Get Yourself Some Gin.* Homepage des Gin-Webframeworks. URL: https://github.com/gin-gonic/gin (besucht am 01. 12. 2024).

[74] *Stimulus Reference*. Dokumentation über die Stimulus Element values. URL: https://stimulus.hotwired.dev/reference/values (besucht am 01. 12. 2024).

[75] *Visjs/Vis-Network*. vis.js, 1. Dez. 2024. URL: https://github.com/visjs/vis-network (besucht am 01. 12. 2024).

[76] *$sort (Aggregation) - MongoDB Manual v8.0*. Dokumentation der MongoDB Sortierfunktion. URL: https://www.mongodb.com/docs/current/reference/operator/aggregation/sort/ (besucht am 10. 12. 2024).

[77] Jurek Jesse. *Dumpeldown/Acema_ oran at Dev*. Repository des verbesserten ACEMA Quellcodes. URL: https://github.com/dumpeldown/acema_oran/tree/dev (besucht am 26. 11. 2024).

[78] *Bash/Cmd inside Container - Threat Matrix for Kubernetes*. Beschreibung der Technik T1059. URL: https://microsoft.github.io/Threat-Matrix-for-Kubernetes/techniques/bash%20or%20cmd%20inside%20container/ (besucht am 13. 12. 2024).

[79] *Container Service Account - Threat Matrix for Kubernetes*. Beschreibung der Technik T1528. URL: https://microsoft.github.io/Threat-Matrix-for-Kubernetes/techniques/container%20service%20account/ (besucht am 13. 12. 2024).

[80] *CVE Landing – Mappings Explorer*. URL: https://center-for-threat-informed-defense.github.io/mappings-explorer/external/cve/all-data.html (besucht am 13. 12. 2024).

[81] *Home – Mappings Explorer*. Homepage des Mappings Explorers. URL: https://center-for-threat-informed-defense.github.io/mappings-explorer/ (besucht am 14. 12. 2024).

[82] *Vulcan*. Homepage des Vulcan MITRE Mappers. URL: https://vulcan.io/voyager18/mitre-mapper/ (besucht am 13. 12. 2024).

[83] Ortal Keizman. *The MITRE ATT&CK Framework and More - Introducing Voyager18*. Ankündigung der Veröffentlichung des Vulcan MITRE Mappers. 4. Juli 2022. URL: https://vulcan.io/blog/the-mitre-attck-framework-and-more-introducing-voyager18/ (besucht am 14. 12. 2024).

[84] *NVD Program Announcement - Updated*. URL: https://nvd.nist.gov/general/news/nvd-program-transition-announcement (besucht am 24. 11. 2024).

[85] *National Vulnerability Database*. Homepage der NIST NVD. 17. Mai 2024. URL: https://www.nist.gov/itl/nvd (besucht am 24. 11. 2024).

[86] Julia Wunder, Alan Corona, Andreas Hammer und Zinaida Benenson. „On NVD Users' Attitudes, Experiences, Hopes, and Hurdles". In: *Digital Threats* 5.3 (26. Okt. 2024), 33:1–33:19. DOI: 10.1145/3688806. URL: https://dl.acm.org/doi/10.1145/3688806 (besucht am 24. 11. 2024).

[87] Yossi Weizmann und Dotan Patrich. *Mitigate Threats with the New Threat Matrix for Kubernetes*. Letztes Update über die Threat Matrix für Kubernetes. 7. Dez. 2022. URL: https://www.microsoft.com/en-us/security/blog/2022/12/07/mitigate-threats-with-the-new-threat-matrix-for-kubernetes/ (besucht am 10. 12. 2024).

[88] *Official CVE Feed*. Veröffentliche CVE im direkten Zusammenhng mit Kubernetes. URL: https://kubernetes.io/docs/reference/issues-security/official-cve-feed/ (besucht am 10. 12. 2024).

[89] *You Searched for Threat Matrix for Kubernetes*. Suche nach Blogeinträgen über die Threat Matrix von Microsoft. 7. Dez. 2022. URL: https://www.microsoft.com/en-us/security/blog/search/threat%20matrix%20for%20kubernetes/ (besucht am 10. 12. 2024).

[90] *Acema_ oran/OCloud_ Data_ Gathering.Py at Master · Fklement/Acema_ oran*. Quellcode einer Datei im ACEMA Repository. URL: https://github.com/fklement/acema_oran/blob/master/OCloud_Data_Gathering.py (besucht am 21. 11. 2024).

[91] *NVD - CVSS v4.0 Official Support*. URL: https://nvd.nist.gov/general/news/cvss-v4-0-official-support (besucht am 21. 11. 2024).

[92] *Retirement of CVSS V2*. URL: https://nvd.nist.gov/general/news/retire-cvss-v2 (besucht am 21. 11. 2024).

[93] *Cti/USAGE-CAPEC.Md at Master · Mitre/Cti*. Anleitung zur Verwendung der CTI-Daten. URL: https://github.com/mitre/cti/blob/master/USAGE-CAPEC.md (besucht am 24. 11. 2024).

[94] Dave Dugal. *Announcing CVSS v4.0*. URL: https://www.first.org/cvss/v4-0/cvss-v40-presentation.pdf (besucht am 14. 12. 2024).

[95] *NVD - Vulnerability Metrics*. URL: https://nvd.nist.gov/vuln-metrics/cvss (besucht am 21. 11. 2024).

Anhang

A. Quellcodes zur Arbeit

In der Arbeit werden verschiedene Quellcodes referenziert. Es folgt eine Auflisting der Quellcodes, die in der Arbeit verwendet wurden, und wo sie zu finden sind.

- **Quellcode des Dashboards**: Der Quellcode des Dashboards ist in einem Git-Repository auf GitLab verfügbar. Dieselbe Gitlab-Instanz enthält auch die Projekte und Quellcodes für andere Komponenten des Forschungsprojekts. Zum aktuellen Zeitpunkt ist der Zugriff auf das Repository nur für autorisierte Personen möglich. Das Forschungsprojekt verfolgt das Ziel, alle Quellcodes nach Abschluss des Projekts zu veröffentlichen, voraussichtlich im ersten Halbjahr des Jahres 2025. Der genaue Link zum Repository wird zu gegebener Zeit unter `https://www.5g-foran.com/` veröffentlicht.
- **Quellcode von Klement, Liu und Katzenbeisser**: Der Quellcode der Implementation von Klement, Liu und Katzenbeisser ist in einem Git-Repository auf GitHub verfügbar. Im Fließtext dieser Arbeit wird dieser Quelltext als *originaler Quellcode* bezeichnet. Der Quellcode ist unter folgendem Link verfügbar: `https://github.com/fklement/acema_oran`
- **Fork des Quellcodes von Klement, Liu und Katzenbeisser**: Der Quellcode des Forks von Klement, Liu und Katzenbeisser ist in einem Git-Repository auf GitHub verfügbar. Der Fork enthält die eigens vorgenommenen Änderungen und wird im Fließtext dieser Arbeit als *verbesserter Quellcode* bezeichnet. Der Quellcode ist unter folgendem Link verfügbar: `https://github.com/dumpeldown/acema_oran`

B. Ausschnitt aus dem ACEMA-Output

```
{
 "scan_date": "2024-12-10",
 "scan_runtime": "00h 38m and 35.43s",
 "data": [
  {
   "technique_id": "T1485",
   "t_findings": []
  },
  {
   "technique_id": "T1070",
   "t_findings": [
    {
     "capec_id": "CAPEC-268",
     "c_findings": [
      {
       "cwe": "CWE-117",
       "cves": [
        {
         "cve_id": "CVE-2006-4624",
         "score": [
          "V2",
          2.6,
          "LOW"
         ],
         "v2_score": 2.6,
         "v2_exploitability_score": 4.9,
         "v2_impact_score": 2.9,
         "v2_vector": "AV:N/AC:H/Au:N/C:N/I:P/A:N",
         "access_vector": "NETWORK",
         "full_metrics": [
          {
           "source": "nvd@nist.gov",
           "type": "Primary",
           "cvssData": {
            "version": "2.0",
            "vectorString": "AV:N/AC:H/Au:N/C:N/I:P/A:N",
            "baseScore": 2.6,
            "accessVector": "NETWORK",
            "accessComplexity": "HIGH",
            "authentication": "NONE",
```

```
        "confidentialityImpact": "NONE",
        "integrityImpact": "PARTIAL",
        "availabilityImpact": "NONE"
       },
       "baseSeverity": "LOW",
       "exploitabilityScore": 4.9,
       "impactScore": 2.9,
       "acInsufInfo": false,
       "obtainAllPrivilege": false,
       "obtainUserPrivilege": false,
       "obtainOtherPrivilege": false,
       "userInteractionRequired": true
      }
     ],
     "description": "CRLF injection vulnerability in...",
     "cpe_vulnerable": true,
     "cpe_criteria": "cpe:2.3:a:gnu:mailman:*:*:*:*:*:*...",
     "published": "2006-09-07T19:04:00.000",
     "last_modified": "2024-11-21T00:16:24.163"
    }
   ]
  }
 ]
},
```

C. Ausschnitt aus dem angepassten ACEMA-Output

```
{
 "data": [
  {
   "technique_id": "T1485",
   "t_findings": [],
   "avg_score": 0,
   "avg_impact_score": 0,
   "avg_exploitability_score": 0
  },
  {
   "technique_id": "T1070",
   "t_findings": [
    {
     "capec_id": "CAPEC-268",
     "c_findings": [
      {
       "cwe": "CWE-117",
       "cves": [
        {
         "id": "CVE-2006-4624",
         "v2_score": 2.6,
         "v2_impact_score": 2.9,
         "v2_exploitability_score": 4.9
        }
       ]
      }
     ]
    },
```

D. Mapping Datensatz

Tabelle D.1.: Mapping von Threat-ID zu MITRE-Technik und MITRE-Taktik

Threat-ID	TK4M Technique Name	MITRE-Technique	MITRE-Taktik
T-IMG-01	Application Vulnerability	T1190	initial-access
T-VM-C-01	bash or cmd inside container	T1059	execution
T-VM-C-01	bash or cmd inside container	T1204	execution
T-ADMIN-02	bash or cmd inside container	T1059	execution
T-ADMIN-02	bash or cmd inside container	T1204	execution
T-HW-01	bash or cmd inside container	T1059	execution
T-HW-01	bash or cmd inside container	T1204	execution
T-IMG-01	bash or cmd inside container	T1059	execution
T-IMG-01	bash or cmd inside container	T1204	execution
T-VM-C-01	Kubernetes CronJob	T1053.007	persistence
T-VM-C-01	Clear container logs	T1070	defense-evasion
T-IMG-01	Compromised Image In Registry	T1195.002	initial-access
T-IMG-01	Compromised Image In Registry	T1525	initial-access
T-GEN-02	Access Managed Identity credentials	T1552	credential-access
T-O2-01	Network mapping	T1046	discovery
T-VM-C-02	Cluster internal networking	T1210	lateral-movement
T-IMG-01	images from a private registry	T1530	collection
T-IMG-04	images from a private registry	T1530	collection
T-ADMIN-02	Exec into container	T1609	execution
T-O2-01	Exec into container	T1609	execution
T-VM-C-03	Exec into container	T1609	execution
T-VM-C-04	Exec into container	T1609	execution
T-VM-C-01	Backdoor container	T1543	persistence
T-VM-C-01	Backdoor container	T1542	persistence
T-VL-02	Backdoor container	T1543	persistence
T-VL-02	Backdoor container	T1542	persistence
T-GEN-02	Mount service principal	T1552	credential-access
T-O2-01	Access Kubelet API	T1613	discovery
T-IMG-03	List K8S secrets	T1552	credential-access
T-VM-C-04	Resource hijacking	T1496	impact
T-VL-01	Resource hijacking	T1496	impact
T-O2-01	Delete K8S events	T1070	defense-evasion
T-VL-01	Pod or container name similarily	T1036	defense-evasion
T-VL-01	Pod or container name similarily	T1036.005	defense-evasion
T-IMG-01	Pod or container name similarily	T1036	defense-evasion
T-IMG-01	Pod or container name similarily	T1036.005	defense-evasion
T-ADMIN-01	Denial of service	T1498	impact

T-ADMIN-01	Denial of service	T1499	impact
T-VM-C-04	Denial of service	T1498	impact
T-VM-C-04	Denial of service	T1499	impact
T-VM-C-05	Denial of service	T1498	impact
T-VM-C-05	Denial of service	T1499	impact
T-GEN-02	container service account	T1528	credential-access
T-GEN-02	container service account	T1528	lateral-movement
T-GEN-02	container service account	T1528	persistence
T-ADMIN-02	Application credentials in configuration files	T1552	credential-access
T-VM-C-03	Application credentials in configuration files	T1552	credential-access
T-ADMIN-02	Application credentials in configuration files	T1552	lateral-movement
T-VM-C-03	Application credentials in configuration files	T1552	lateral-movement
T-VM-C-03	Instance Metadata API	T1552	discovery
T-IMG-03	Instance Metadata API	T1552	discovery
T-VM-C-01	Data destruction	T1485	impact
T-GEN-01	Application exploit (RCE)	T1190	execution
T-VM-C-01	Privileged container	T1610	privilege-escalation
T-VM-C-02	Writable hostPath mount	T1611	persistence
T-VM-C-02	Writable hostPath mount	T1611	privilege-escalation
T-VM-C-02	Writable hostPath mount	T1611	lateral-movement
T-O2-01	Access the K8S API server	T1613	discovery
T-GEN-02	Exposed sensitive interfaces	T1133	initial-access
T-GEN-02	Exposed sensitive interfaces	T1078	initial-access
T-GEN-02	Exposed sensitive interfaces	T1133	discovery
T-GEN-02	Exposed sensitive interfaces	T1078	discovery
T-IMG-01	New Container	T1610	execution
T-IMG-01	New Container	T1612	execution
T-IMG-04	New Container	T1610	execution
T-IMG-04	New Container	T1612	execution

E. Liste der Routen im Dashboard

- / **(Startseite):** Statische Startseite des Dashboards mit Buttons zur Navigation zu anderen Routen.
- **/acema (Darstellung der ACEMA Diagramme):** Enthält alle über ACEMA erstellen Diagramme als PDFs.
- **/mitre/kubernetes_matrix (Kubernetes Matrix):** Visualisierung der angepassten MITRE ATT&CK Matrix.
  - **Dynamisch erzeugt aus:**
    - **kubernetes_matrix.tmpl:** Nutzt die Anzahl der Taktiken und Techniken zum Aufbau der Zeilen und Spalten.
    - **technique.partial.tmpl:** Füllt die Felder mit den in `pkg/mitre/technique.go` definierten Techniken.
- **/artifacts (Artefakte - Index):** Liefert eine Liste aller Artefakte, aller Filtermöglichkeiten und einer visuellen Darstellung auf der Zeitachse.
  - **Query-Parameter (optional):**
    - `tool_name` - Filtert Artefakte nach dem Namen des Tools.
    - `tactic` - Filtert Artefakte nach der zugehörigen Taktik.
    - `microsoft_technique` - Filtert Artefakte nach Microsoft-Techniken.
    - `start_date` - Filtert Artefakte nach Startzeitpunk.
    - `end_date` - Filtert Artefakte nach Endzeitpunkt.
  - **Dynamisch erzeugt aus:**
    - **artifacts/index.tmpl:** Nutzt die Daten aller Artefakte und die Werte der Filterparameter.
- **/artifacts/:id (Artefakte - Details):** Liefert Details zu einem spezifischen Artefakt anhand der `id`.
  - **Dynamisch erzeugt aus:**

        * **artifacts/show.tmpl:** Nutzt die Daten eines Artefaktes.
        * **error.tmpl:** Spezifische Fehlermeldung von Go nach Datenbankabfrage.

- **/campaigns (Kampagnen - Index):** Liefert eine Liste aller Kampagnen.
  - **Dynamisch erzeugt aus:**
    * **campaigns/index.tmpl:** Nutzt die Daten aller Artefakte und die Werte der Filterparameter.
- **/campaigns/:id (Kampagnen - Details):** Liefert Details zu einer spezifischen Kampagne anhand der `id`.
  - **Query-Parameter (optional):**
    * `start_date` - Filtert Kampagnen nach Startzeitpunk.
    * `end_date` - Filtert Kampagnen nach Endzeitpunkt.
  - **Dynamisch erzeugt aus:**
    * **campaigns/show.tmpl:** Nutzt die Daten einer Kampagne.
    * **error.tmpl:** Spezifische Fehlermeldung von Go nach Datenbankabfrage.

F. Datenstruktur

Abbildung F.1.: Struktur eines Artefakts in der Datenbank und in Go.

G. Angepasste Matrix im Dashboard

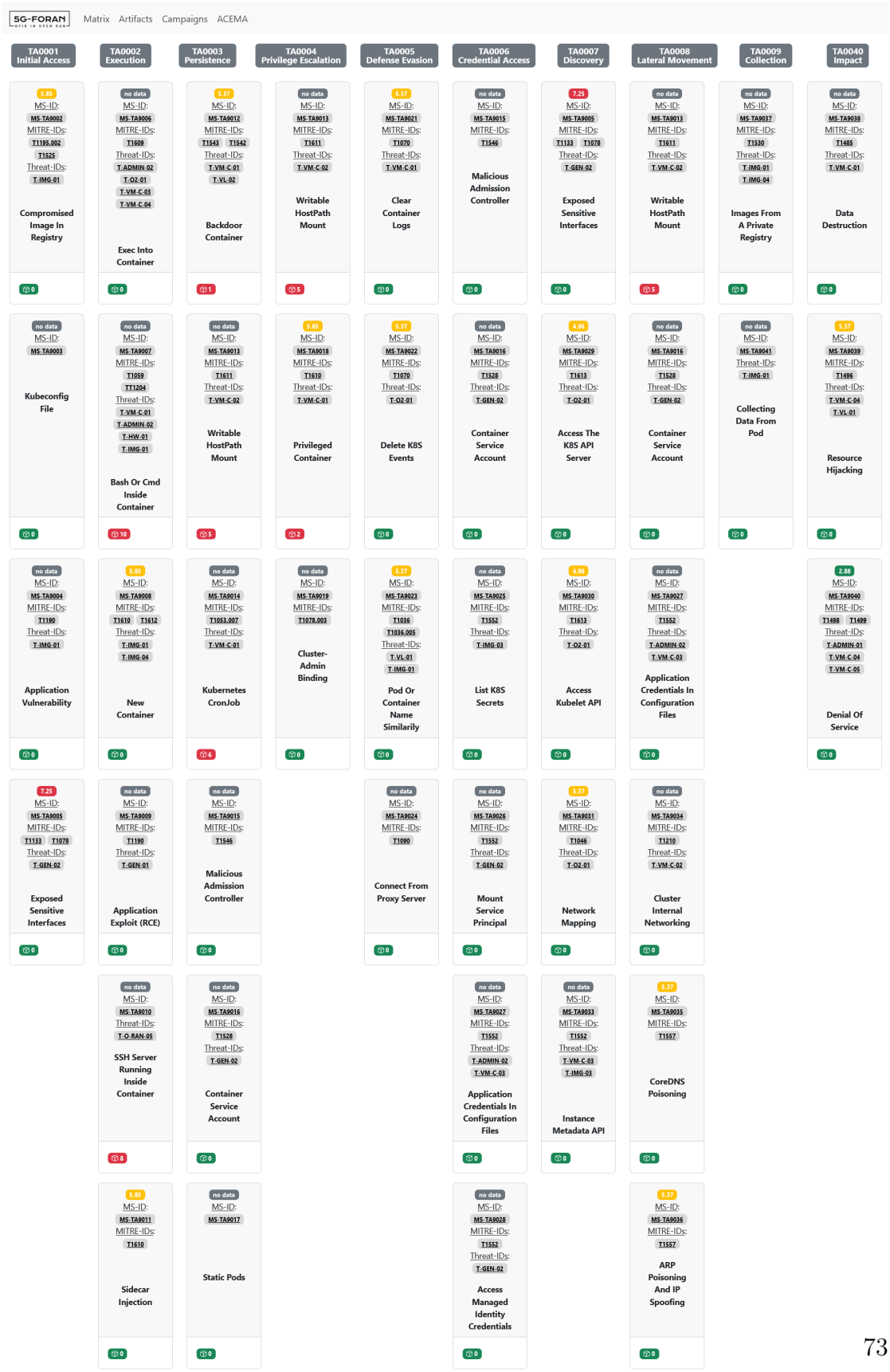

Abbildung G.1.: Darstellung aller Techniken in der Matrix.

H. Diagramme aus ACEMA

Die Werte der Metriken in einem CVSSVersion 2 Vektor werden typischerweise nicht numerisch, sondern textuell ausgedrückt. Um mit den Werten zu rechnen, wird die textuelle Darstellung über Formeln in einen numerischen Wert umgewandelt, die in der CVSSVersion 2 Spezifikation hinterlegt sind [65]. Diese Werte werden in der ACEMA Implementation genutzt, um die Werte in einem Spektrum zwischen 0 und 1 darzustellen, um diese in einem Netzdiagramm darzustellen (vgl. Abbildung H.1).

$$\text{AccessVector} = \begin{cases} \text{requires local access} & : 0{,}395 \\ \text{adjacent network accessible} & : 0{,}646 \\ \text{network accessible} & : 1{,}0 \end{cases}$$

$$\text{AccessComplexity} = \begin{cases} \text{high} & : 0{,}35 \\ \text{medium} & : 0{,}61 \\ \text{low} & : 0{,}71 \end{cases}$$

$$\text{Authentication} = \begin{cases} \text{requires multiple instances of authentication} & : 0{,}45 \\ \text{requires single instance of authentication} & : 0{,}56 \\ \text{requires no authentication} & : 0{,}704 \end{cases}$$

$$\text{ConfImpact} = \begin{cases} \text{none} & : 0{,}0 \\ \text{partial} & : 0{,}275 \\ \text{complete} & : 0{,}660 \end{cases}$$

$$\text{IntegImpact} = \begin{cases} \text{none} & : 0{,}0 \\ \text{partial} & : 0{,}275 \\ \text{complete} & : 0{,}660 \end{cases}$$

$$\text{AvailImpact} = \begin{cases} \text{none} & : 0{,}0 \\ \text{partial} & : 0{,}275 \\ \text{complete} & : 0{,}660 \end{cases}$$

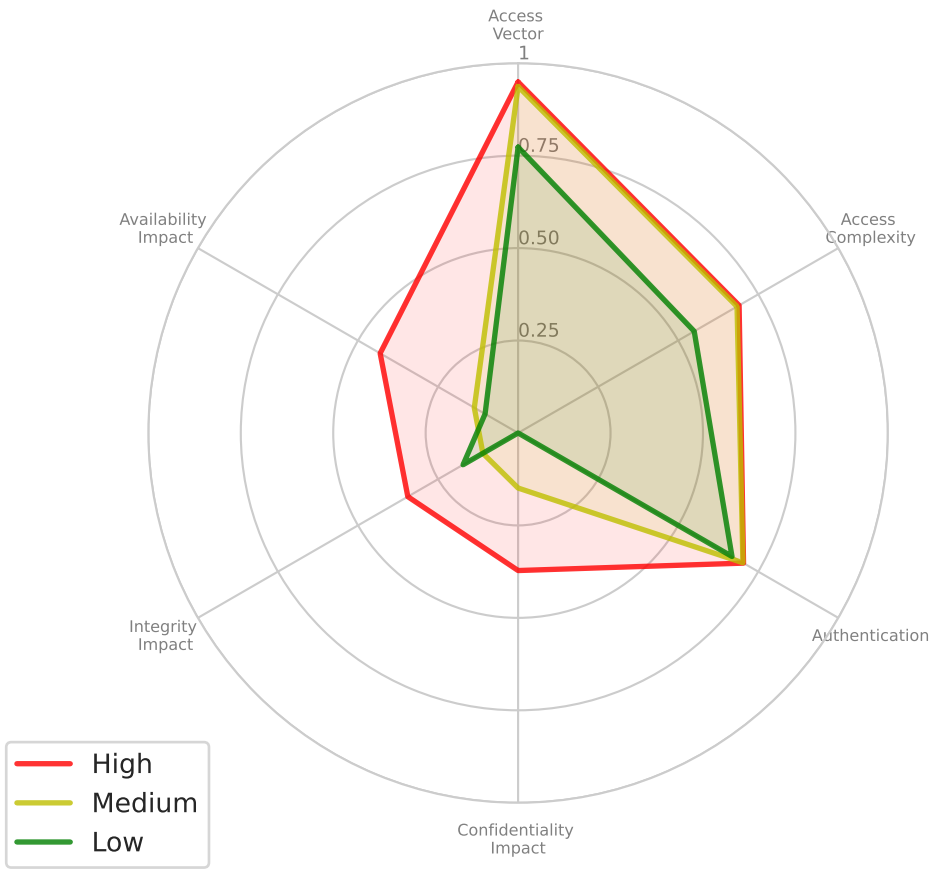

Abbildung H.1.: Durchschnittlicher CVSS Vektor aller CVEs pro Schweregrad

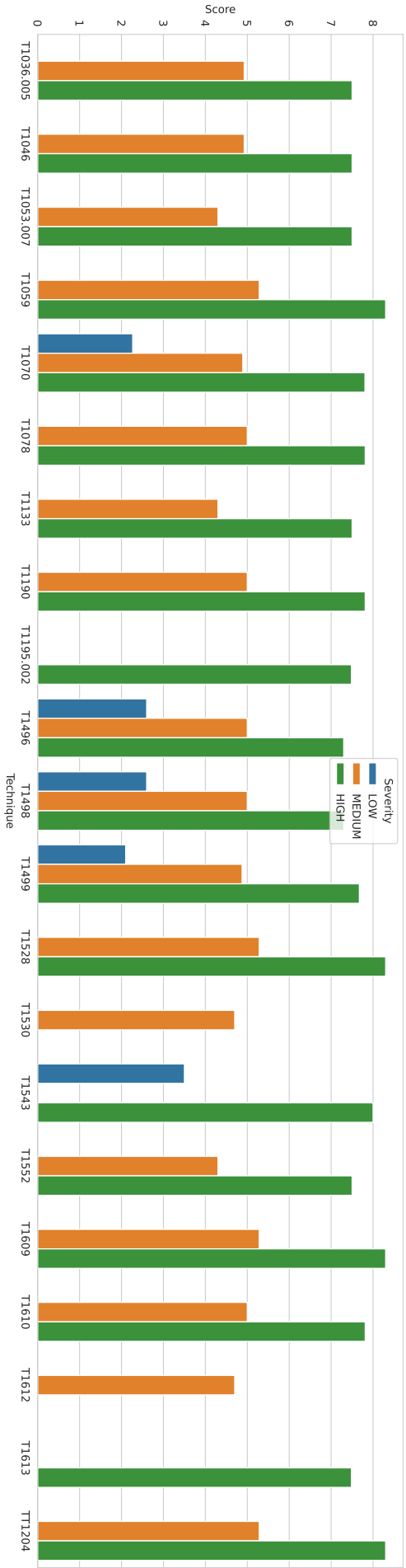

Abbildung H.2.: Durchschnitt aller CVSS Werte der CVEs pro MITRE-Technik und Schweregrad

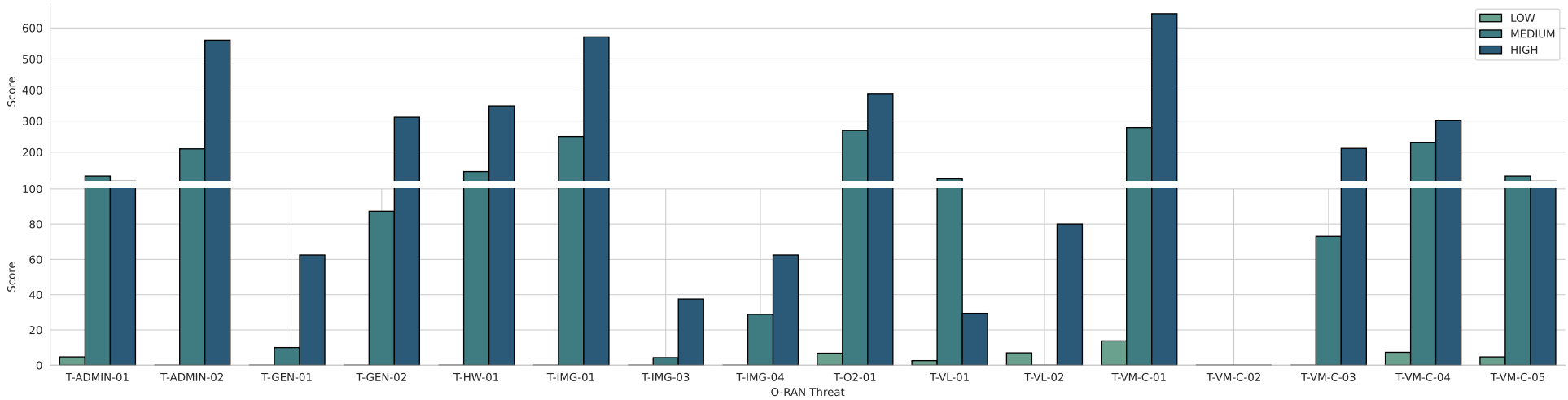

Abbildung H.3.: Summierung aller CVSS Werte der CVEs pro O-RAN Threat-ID und Schweregrad

Erklärung

Ich versichere, die von mir vorgelegte Arbeit selbstständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer oder der Verfasserin/des Verfassers selbst entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

Wiesbaden, 17.12.2024

Ort, Datum

Unterschrift