HAW HAMBURG

Masterarbeit

Peer Ricklev Maute

Automatisierung von Incident-Response-Prozessen

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department Computer Science

Peer Ricklev Maute

Automatisierung von Incident-Response-Prozessen

Masterarbeit eingereicht im Rahmen der Masterprüfung
im Studiengang *Master of Science Informatik*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Klaus-Peter Kossakowski
Zweitgutachterin: Prof. Dr. Bettina Buth

Eingereicht am: 06. Dezember 2024

Peer Ricklev Maute

Thema der Arbeit

Automatisierung von Incident-Response-Prozessen

Stichworte

Incident Response, Cybersecurity, SOAR, Automatisierung

Kurzzusammenfassung

Das Ziel dieser Arbeit ist es, darzulegen, wie eine Automatisierung von Incident-Response-Prozessen möglich sein kann. Dies wird beispielhaft an den Prozessen der HAW Hamburg aufgezeigt. Hierzu wurden eingehende Warnmeldungen über einen Zeitraum von etwa acht Monaten betrachtet. Für diese Warnmeldungen wurden Filteroptionen konzipiert, die irrelevante Meldungen herausfiltern und die übrigen Meldungen priorisieren. Für die restlichen Warnmeldungen wurden zudem Prozessdiagramme erstellt, die die nötigen Reaktionen auf Warnmeldungen zeigen und sich an den vor dieser Arbeit bestehenden, generischen Incident-Handling-Prozess der HAW Hamburg anschließen und diesen erweitern. Die Dezimierung an manuell zu bearbeitenden Warnmeldungen, die sich aus den erarbeiteten Filtern ergibt, birgt derartig weitreichende Implikationen, dass aus einer erfolgreichen Implementierung der vorgeschlagenen Filter eine komplette Transformation des zugrundeliegenden Prozesses resultiert.

Peer Ricklev Maute

Title of Thesis

Automation of Incident Response Processes

Keywords

Incident Response, Cybersecurity, SOAR, Automation

Abstract

The aim of this thesis is to show how the automation of incident response processes can be approached. This is demonstrated using the processes of the HAW Hamburg as an example. This involved examining the incoming alerts spanning a period of eight months. Filter options were developed for these alerts to filter out irrelevant alerts and prioritize the remaining alerts. For these remaining alerts, process diagrams were also created that show the necessary reactions to alerts and are aligned with and extend the existing generic incident handling process of the HAW Hamburg that existed prior to this work. The decimation of manually processed alerts resulting from the developed filters has such far-ranging implications that a successful implementation of the proposed filters would result in a complete transformation of the underlying process.

Inhaltsverzeichnis

Abbildungsverzeichnis **viii**

Tabellenverzeichnis **x**

1 Einleitung **1**
1.1 Motivation und Problemstellung 1
1.2 Ziel der Arbeit 2
1.3 Aufbau 3

2 Grundlagen **5**
2.1 Incident Response Grundbegriffe 5
2.1.1 Incident Response 6
2.1.2 Incident Handling 6
2.1.3 Teamtypen 7
2.1.4 Security Information and Event Management (SIEM) 8
2.1.5 Security Orchestration Automation and Response (SOAR) 9
2.2 Playbooks 9
2.2.1 Aufsetzen von Playbooks 10
2.2.2 Aufbau und Datenformate 11
2.3 Security Information Sharing 13
2.3.1 MISP 13
2.3.2 Automatische Warnmeldungen 14

3 Daten **16**
3.1 Stand der Dinge 16
3.1.1 Datenformat 16
3.1.2 Überblick über bisherige Filter 18
3.2 Weitere Datenanalyse 19
3.2.1 Ein genauer Blick auf die Daten 20

3.2.2 Neue Filtermöglichkeiten . . . 24
3.2.3 Filterkategorien . . . 25
3.3 Gefilterte Daten . . . 26
3.3.1 Filtereffektivitäten . . . 27
3.3.2 Resultierende Daten . . . 28

4 Playbooks **31**
4.1 Playbook-Konzepte . . . 32
4.1.1 Playbook-Nummer 1.0.0 - Main-Playbook für AW-Messages . . . 32
4.1.2 Playbook-Nummer 1.1.0 - Sub-Playbook: Bot . . . 33
4.1.3 Playbook-Nummer 1.2.0 - Sub-Playbook: Vulnerability . . . 35
4.1.4 Playbook-Nummer 1.3.0 - Sub-Playbook: Security Monitoring/Blocklist . . . 36
4.1.5 Playbook-Nummer 1.4.0 - Sub-Playbook: Configuration . . . 36
4.2 Playbook-Prozesse . . . 38
4.2.1 Generischer Incident-Handling-Prozess . . . 38
4.2.2 Spezifische Playbooks . . . 41

5 Von der Theorie zur Praxis **54**
5.1 Auswirkungen . . . 54
5.1.1 Berechnung . . . 56
5.1.2 Ergebnis . . . 57
5.2 Praktische Umsetzung der Automatisierung . . . 58
5.3 Erster Implementierungsvorschlag . . . 60
5.3.1 Architektur . . . 61
5.3.2 Wahl des Ticketing-Systems . . . 63
5.3.3 Nachtrag zum Implementierungsvorschlag . . . 64
5.4 Kritik an SOAR-Tools . . . 65

6 Fazit **67**
6.1 Erkenntnisse . . . 67
6.2 Limitationen . . . 68
6.3 Ausblick . . . 69

Literaturverzeichnis **72**

A Anhang **75**

A.1 Prozessdiagramme 75

A.2 XML-Schema-Datei für Warnmeldungen 80

A.3 Verwendete Hilfsmittel 83

Selbstständigkeitserklärung **84**

Abbildungsverzeichnis

2.1 Übersicht der möglichen Services eines CSIRTs - aus [7] 7
2.2 Nutzung von vorgefertigten Playbooks - aus [22] 10
2.3 Analyseprozess für AW-Meldungen - aus [6] 15

3.1 Beispiel einer AW-Meldung, aus [15] 17
3.2 Sankey-Diagramm der Filterwellen, aus [15] 19
3.3 Zeitstrahl der gefilterten Meldungen 20
3.4 Meldungsanzahl nach Wochentagen 22
3.5 Anonymisierter Ausschnitt der Security Monitoring/Blocklist Wiederholungsmeldungen 23
3.6 Anzahl der herausgefilterten Meldungen pro Filter (alleinstehend ausgeführt) 27
3.7 Sequenzielle Anwendung der Filterkategorien 28
3.8 Anteil der Meldungstypen an den weiter zu bearbeitenden Warnmeldungen 29

4.1 Phasen des generischen Incident-Handling-Prozesses der HAW Hamburg - eigene Darstellung nach [12] 39
4.2 Übersicht der Rollen im generischen Incident-Handling-Prozess der HAW Hamburg - aus [12] 41
4.3 Prozess: Detect and Report - verändertes Diagramm aus [12] 43
4.4 Prozess: Bot 47
4.5 Prozess: Vulnerability 49
4.6 Prozess: Security Monitoring/Blocklist 51
4.7 Ausschnitt Prozess: Configuration 52

5.1 Ursprünglicher Prozess zur Bearbeitung von Warnmeldungen - aus [12] . . 55
5.2 Möglicher Aufbau der Implementierung 61
5.3 Beispiel für ein Ticket in OpenProject - aus [4] 64

A.1 Prozessdiagramm: Detect and Report 75
A.2 Prozessdiagramm: Bot 76

A.3 Prozessdiagramm: Vulnerability 77
A.4 Prozessdiagramm: Security Monitoring/Blocklist 78
A.5 Prozessdiagramm: Configuration 79

Tabellenverzeichnis

A.1 Verwendete Hilfsmittel und Werkzeuge 83

1 Einleitung

Die Weihnachtszeit ist für viele Menschen eine Zeit der Ruhe und Besinnlichkeit. Auch an der Hochschule für Angewandte Wissenschaften (HAW) Hamburg sind die Weihnachtsferien für gewöhnlich ein letztes Kraftschöpfen vor dem Endspurt des Wintersemesters im neuen Jahr. Am 29. Dezember 2022 wurde jedoch ein Angriff auf die IT-Infrastruktur der Hochschule festgestellt, der weitreichende Konsequenzen nach sich zog [1].

Dieser Vorfall ist kein Einzelfall. So essenziell wie die IT-Infrastruktur für alle Lebensbereiche heute ist, ergeben sich aus dieser kritischen Position heraus auch neue Risiken. Auf der anderen Seite professionalisiert sich die Hacker-Szene zunehmend. Konnte man früher noch für viele Hacking als eine Art strafbares Hobby bezeichnen, erschließen sich heute aus Cyberangriffen komplett neue Geschäftsfelder, wie das Beispiel *Ransomware-as-a-Service* zeigt. Hierbei werden im Darknet Cyberangriffe mit Ransomware mittlerweile als eigener Dienst angeboten [17].

1.1 Motivation und Problemstellung

Da sich die Cyberkriminalität zunehmend professionalisiert, muss sich auch die Cybersicherheit entsprechend anpassen. Ein entscheidender Stichpunkt, der in diesem Zusammenhang häufig fällt, ist die Automatisierung. In der Vergangenheit wurden bei der Incident Response, also der Reaktion auf einen Sicherheitsvorfall, viele Aufgaben manuell abgearbeitet. Hieraus ergeben sich eine Reihe an Optimierungsmöglichkeiten. Zunächst gibt es ein generelles Einschleichen von Fehlern bei menschlichem Handeln. Entscheidend sind aber auch die Mean-Time-to-Detect (MTTD) und die Mean-Time-to-React (MTTR). Hierbei handelt es sich um die durchschnittliche Zeit, die es dauert, einen Vorfall zu erkennen und im Anschluss darauf zu reagieren. Eine Automatisierung dieser Prozesse führt zu einer immensen Zeitersparnis in diesen kritischen Bereichen [16].

Die Vorteile der Automatisierung von Incident-Response-Prozessen sind zwar klar ersichtlich, in der Praxis stellt diese Automatisierung viele Organisationen jedoch vor eine erhebliche Herausforderung. Die Gründe hierfür können vielseitig sein. Im Zweifelsfall sind die bestehenden Strukturen allerdings organisch gewachsen und die Prozesse somit recht starr. Entsprechend wurden sie auch nicht für eine derartige Umstrukturierung ausgelegt.

Eine Organisation, die nicht zuletzt wegen dem Cyberangriff im Jahre 2022 ein gesteigertes Interesse an der Optimierung ihrer Incident-Response-Prozesse hat, ist die HAW Hamburg. Zu diesem Thema wurde bereits eine Sammlung an Forschungsarbeiten verfasst. Auch diese Arbeit bettet sich ein in eine Reihe an Arbeiten, die sich zum einen mit der Automatisierung von Incident-Response-Prozessen generell und auch spezifisch mit Prozessen an der HAW Hamburg beschäftigen. In [15] wurde bereits einige Vorarbeit geleistet, die den Prozess von automatischen Warnmeldungen betrachtet, die bei der HAW Hamburg eingehen und wie Filtermöglichkeiten kreiert werden können, um die Vielzahl an irrelevanten Warnmeldungen herauszusieben und die übrigen Meldungen zu priorisieren. Es wurden zudem die resultierenden Prozesse betrachtet, die ausgelöst werden, wenn eine relevante Warnmeldung eingeht. Die Arbeit weist jedoch deutliches Verbesserungspotenzial auf. Zum einen sind die Filter noch nicht effektiv genug und diverse irrelevante Warnmeldungen werden nicht herausgefiltert. Zum anderen werden die resultierenden Prozesse lediglich auf sehr konzeptueller und abstrakter Ebene umrissen und noch nicht in den generischen Prozess eingebettet, der bereits klar definiert in [12] vorliegt.

1.2 Ziel der Arbeit

Ein erklärtes Ziel dieser Arbeit ist es, die oben vorgestellten Lücken in [15] zu schließen - also sowohl eine Optimierung der bestehenden Filter vorzunehmen als auch eine genauere Darstellung der resultierenden Prozesse zu erarbeiten, die zudem in die bestehenden Darstellungen eingebettet werden sollen. Zunächst sollen hierzu die Filtermöglichkeiten verfeinert werden, um die Anzahl an Warnmeldungen, die automatisiert herausgefiltert werden können, zu maximieren. Es soll zudem eine einheitliche Definition aller resultierenden Prozesse geben, die die bereits bestehende Definition des generischen Prozesses um die spezifischeren Reaktionen ergänzt. Wie oben beschrieben, stellt sich die Frage, wie man die Automatisierung von Incident-Response-Prozessen praktisch angeht, für eine

Vielzahl an Organisationen. Greifbare Beispiele für eine solche Transformation sind somit entscheidend, um derartige Automatisierungsbestrebungen weiter voranzutreiben.

Aufgrund seiner Anschaulichkeit steht das Ziel im Fokus der vorliegenden Arbeit. Im Grunde ist es dennoch nur ein Hilfsmittel für ein weiteres Ziel der Arbeit. Dieses ist es, an einer Art Fallbeispiel aufzuzeigen, wie bestehende Incident-Response-Prozesse automatisiert werden können. Es soll somit nicht nur das konkrete Beispiel der HAW Hamburg mit ihren Prozessen betrachtet werden, sondern es muss auch vor dem größeren Hintergrund der generellen Automatisierung von Incident-Response-Prozessen geschehen.

1.3 Aufbau

Zu Beginn ist es notwendig, einige Grundbegrifflichkeiten zu klären, die für den Rahmen dieser Arbeit entscheidend sind. Hierzu werden zunächst einige Kernkonzepte aus dem Feld der Incident Response vorgestellt. Nicht zuletzt zählt hierzu die Definition des Begriffs selbst und seine Abgrenzung zu dem Begriff Incident Handling, welche beide von zentraler Bedeutung für diese Arbeit sind. Des Weiteren werden mit SIEM und SOAR zwei Technologien betrachtet, die für mögliche Automatisierungsstrategien von Incident-Response-Prozessen in Frage kommen. Inwieweit das Konzept des Playbooks relevant ist, wird ebenfalls besprochen. Zuletzt wird hier noch ein genauerer Blick auf das Feld des Security Information Sharing geworfen und wie sich die automatischen Warnmeldungen, die bei der HAW Hamburg eingehen, hierin einbetten.

Darauf folgt Kapitel 3, das sich mit den Daten beschäftigt. Es wird ein Überblick über den Stand der Dinge aus der Arbeit in [15] geschaffen, worin ebenfalls beschrieben wird, wie die Daten vorliegen und wie diese beschaffen sind. Im Anschluss gibt es dann die weitere Datenanalyse, die neue Filteroptionen für die automatischen Warnmeldungen ergründet. Die resultierenden Daten werden daraufhin genauer untersucht. Es wird besprochen, welche Warnmeldungen letztlich übrig bleiben und wie effektiv die verschiedenen Filter sind.

Kapitel 4 stellt zu Beginn verschiedene Konzepte für Playbooks vor. In diesen Konzepten wird umrissen, welche Reaktionen auf den Eingang einer Warnmeldung von verschiedenen Meldungstypen folgen sollten. Dies schließt zudem an die Arbeit aus [15] an und erweitert die darin vorgestellten Playbook-Konzepte um alle Meldungstypen an Warnmeldungen,

die noch in den zu bearbeitenden Warnmeldungen vorkommen, die nicht durch die Filteroptionen herausgefiltert wurden. Diese Konzepte werden im Anschluss zu tatsächlichen Prozessdiagrammen umgewandelt. Hierzu wird der generische Incident-Handling-Prozess aus [12] betrachtet und um die spezifischen Prozesse ergänzt.

Im Anschluss folgt Kapitel 5, wo ein stärkerer Bogen zur Praxis geschlagen wird. Hier wird zunächst versucht zu ermitteln, welche tatsächlichen Auswirkungen aus den Ergebnissen dieser Arbeit resultieren. Zudem werden Vorschläge für eine praktische Umsetzung der Ergebnisse präsentiert, an die ein kritischer Blick auf die Vorschläge selbst und auf eine Technologie als solches anschließt.

Die Arbeit endet mit einem Fazit, das noch einmal zusammenfasst, was erarbeitet wurde und mit welchen Limitationen diese Arbeit einhergeht, bevor ein Ausblick geboten wird auf zukünftige Arbeiten, die an die Ergebnisse der Arbeit anknüpfen können.

Dieser Aufbau schafft einen klaren Rahmen, um die verschiedenen Aspekte und Herausforderungen, der sich die Automatisierung von Incident-Response-Prozessen gegenübersieht, zu beleuchten.

2 Grundlagen

Dieses Kapitel dient dem Aufarbeiten von Grundlagen, die für die vorliegende Arbeit entscheidend sind. In den späteren Kapiteln wird davon ausgegangen, dass ein Grundwissen im Bereich Informatik vorhanden ist - alle weiteren benötigten Grundlagen für das Verständnis dieser Arbeit werden im folgenden Kapitel erläutert.

2.1 Incident Response Grundbegriffe

Wie in Kapitel 1 bereits umschrieben wurde, handelt es sich bei der vorliegenden Arbeit nicht um ein alleinstehendes Werk. Stattdessen baut sie auf diversen weiteren Arbeiten auf, die in einem gemeinsamen Rahmen verfasst wurden. Hierzu zählen insbesondere [14], [15] und [16]. Diese Arbeiten werden in diesem und auch nachfolgenden Kapiteln mehrfach zitiert. Neben generellen Zitationsgründen gibt eine solche Zitation insbesondere einen Hinweis dafür, dass der besprochene Abschnitt in der genannten Arbeit in größerer Tiefe behandelt wurde.

Hieraus ergibt sich auch eine Art Zwiespalt. Zum einen sollte es nicht notwendig sein, innerhalb einer solch kumulativen Arbeit die gleichen Begriffe in ihrer Ausführlichkeit mehrmals zu definieren, auf der anderen Seite soll die vorliegende Arbeit dennoch unabhängig gelesen werden können. Entsprechend werden nachfolgend einige grundlegende Begriffe aus dem Bereich Incident Response erläutert. Jene Erläuterungen sind bewusst gekürzt und an die anderen Arbeiten angelehnt. Bei Bedarf können die ausführlicheren Beschreibungen in den zitierten Texten nachgelesen werden.

2.1.1 Incident Response

Der Begriff *Incident Response* steht im Hauptfokus dieser Arbeit. Bevor jedoch näher darauf eingegangen werden kann, um was es sich bei einer Incident Response handelt, sollte zunächst klargestellt werden, was überhaupt ein Incident ist.

Ein Incident ist zunächst nichts anderes als ein Geschäftsvorfall. Konkret werden in dieser Arbeit allerdings IT-Security-Incidents betrachtet. Es geht somit um Vorfälle, die die Cybersicherheit einer Organisation bedrohen. Die Reaktionsmaßnahmen, die auf einen solchen Security Incident hin getroffen werden, stellen die *Response* dar. Generell umfasst Incident Response aber jegliche Maßnahmen von Planung und Koordinierung bis hin zur Ausführung, die getroffen werden, um einen Incident abzuschwächen und zu entschärfen. Auch mögliche Handlungen zur Wiederherstellung der Systeme und die Betrachtung der Folgen und Auswirkungen sind durch Incident Response abgedeckt [16].

2.1.2 Incident Handling

Die Begriffe *Incident Response* und *Incident Handling* werden oftmals vermischt. Dies liegt insbesondere daran, dass je nach Organisation die Verantwortlichkeiten, mit denen sich beispielsweise im Rahmen der Incident Response befasst wird, stark variieren können. Wie im vorigen Abschnitt beschrieben, können die Aufgabenbereiche der Incident Response von wenigen Aufgaben für die direkte Reaktion auf einen Sicherheitsvorfall bis hin zur Ausübung präventiver Maßnahmen reichen. Somit ist es nicht weiter verwunderlich, dass die beiden Begriffe mitunter vermischt werden.

In dieser Arbeit wird versucht, einen eindeutigen Unterschied der beiden Begrifflichkeiten zu verwenden. Simpel ausgedrückt, umfasst *Incident Handling* schlicht weitere Handlungen, die über die reine Reaktion auf einen Vorfall (der *Incident Response*) hinausgehen. Hierzu zählen insbesondere die Bereiche *Detect and Reporting*, *Triage* und *Analysis* [2].

Näheres hierzu lässt sich in Kapitel 4 nachlesen, wo detaillierter über die verschiedenen Prozesse des Incident Handling gesprochen wird und diese in Prozessdiagrammen aufgearbeitet werden.

2.1.3 Teamtypen

Für den Rahmen dieser Arbeit sind zudem eine Reihe von Teamtypen relevant, die sich in der Praxis mit Incident-Response-Prozessen beschäftigen. Jeder dieser Teamtypen ist in der Theorie mit einer oder mehrerer Kernaufgaben betraut. In der Realität verschwimmen auch hier die Grenzen zwischen diesen Teams stark und je nach Organisation sind nur ein Teil dieser Teams namentlich vertreten. Auch die verschiedenen Aufgabenbereiche werden in unterschiedlicher Weise über die diversen Teams verteilt. Mit diesen Teamtypen wurde sich umfassender im Rahmen von [16] auseinandergesetzt, der folgende Abschnitt bietet lediglich einen kurzen Überblick.

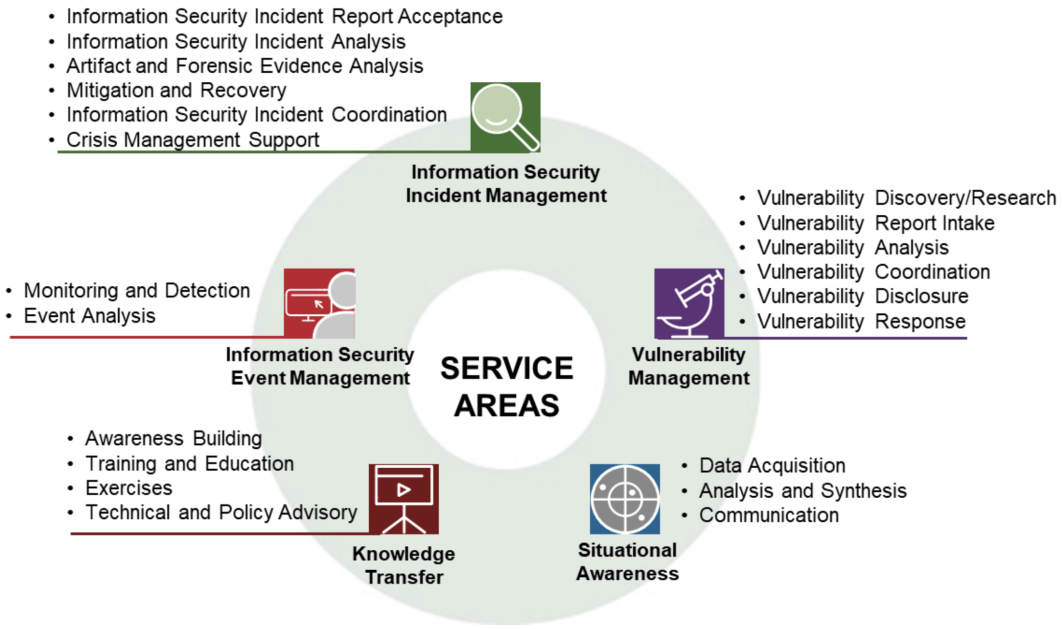

Abbildung 2.1: Übersicht der möglichen Services eines CSIRTs - aus [7]

Ein grober Eindruck der Aufgabenbereiche kann beim Betrachten von Abbildung 2.1 gewonnen werden. Das Forum für Incident Response und Security Teams (FIRST) versucht hier, einen Überblick über die verschiedenen möglichen Dienste eines CSIRTs (Computer Security Incident Response Teams) zu bieten. In der Realität ist die Zuordnung allerdings nicht so deutlich. Tatsächlich gehören die gelisteten Punkte unter Information Security Event Management eher zu den Hauptaufgaben eines Security Operations Centers (SOC). Genauso werden Prozesse aus dem Bereich Situational Awareness häufig von ISACs übernommen (Information Sharing and Analysis Center) [7].

Handelt es sich um eine Organisation, die ein Produkt oder eine Dienstleistung anbietet, gibt es möglicherweise ein Product Security Incident Response Team (PSIRT). Das PSIRT beschäftigt sich in diesem Fall primär mit möglichen Schwachstellen des Produkts oder der Dienstleistung und somit dem Vulnerability Management. Gibt es ein solches PSIRT nicht, müssen Aufgaben wie Vulnerability Coordination oder Vulnerability Response an andere Teams oder Dritte übergeben werden. Im Zweifelsfall wird sich bei Aufgaben wie Vulnerability Discovery auf öffentliche Quellen verlassen und es werden keine gesonderten Prozesse innerhalb der Organisation dafür instanziiert [8].

2.1.4 Security Information and Event Management (SIEM)

Basierend auf bereits existierenden Ansätzen wurde 2005 erstmalig der Begriff Security Information and Event Management (SIEM) definiert. Der Grundgedanke ist hierbei, verschiedene Komponenten zu verbinden, um Security Events von Netzwerkgeräten und Anwendungen zu sammeln und zu organisieren. Durch eine Analyse der Events können SIEM-Tools in Echtzeit potenzielle Bedrohungen und Angriffe erkennen [26].

Über die Jahrzehnte haben sich SIEM-Tools stark gewandelt. Je nach Implementierung gibt es eine Reihe an angebotenen Funktionalitäten. Die Unterschiede etwa für verschiedene Datenquellen oder die Skalierbarkeit sind immens. Auch die Echtzeitverarbeitungsfunktionen und Bedingungen oder Ereignismuster, nach denen festgelegt wird, wann ein Security Incident detektiert wird, unterscheiden sich maßgeblich unter den verschiedenen SIEM-Lösungen. Entsprechend groß sind die Diskrepanzen in der Effektivität bei der Erkennung von potenziellen Bedrohungen.

Der Hauptgrund, warum SIEM-Tools sich so stark in der Cybersecurity-Landschaft verbreitet haben, ist, dass durch den Einsatz von SIEM-Lösungen die MTTD deutlich gesenkt werden kann. Es wird also die Zeit verringert, die verstreicht, bis eine potenzielle Bedrohung identifiziert wird. An dieser Stelle ergibt sich jedoch eine weitere Option, wenn nicht nur die Identifizierung sondern auch die Reaktion automatisiert wird. Obwohl einige aktuelle SIEM-Implementierungen automatisierte Reaktionsmöglichkeiten anbieten, sind diese Funktionalitäten wenn überhaupt sekundär vertreten. Dieses Defizit zu schließen ist Hauptaufgabe des Security Orchestration Automation and Response (SOAR) Ansatzes [9] [16].

2.1.5 Security Orchestration Automation and Response (SOAR)

Nutzt ein CSIRT einen klassischen Techstack ohne SOAR-Tool, beinhalten im Zweifelsfall eine Vielzahl an Aufgaben des CSIRTs viele repetitive Tätigkeiten. Ein Fallstrick ist hierbei, dass viele dieser Prozesse simple Aufgaben darstellen, die stumpf manuell abgearbeitet werden müssen. Nicht nur ist dies zäh und ermüdend, es wächst auch der Spielraum für menschliche Fehler.

Ein weiteres Problem ist, dass der klassische Abarbeitungsprozess eine gewisse Zeit braucht. Eine Vorfallsmeldung wird registriert, dann muss sich eine Person aus dem CSIRT diese durchlesen und entsprechende Schritte einleiten. Gerade bei Cyberangriffen ist die Zeit, die hier zwischen der Identifizierung einer Bedrohung und der Reaktion darauf entscheidend - also die MTTR. Hieraus resultiert ein großer Bedarf für automatisierte Reaktionsmöglichkeiten, was die Kernaufgabe von SOAR-Tools darstellt.

SOAR-Lösungen sind dafür bekannt, bestehende Reaktionsprozesse mit Hilfe von Playbooks zu automatisieren - mehr dazu im Abschnitt 2.2. Anders als SIEM-Tools detektieren SOAR-Tools nicht selbst mögliche Bedrohungen. Stattdessen binden sie Services für die Detektierung an und werden von diesen benachrichtigt, wenn eine Bedrohung vorliegt. Somit wäre es nicht ungewöhnlich, für die Identifizierung von Incidents ein SIEM-Tool an ein SOAR-Tool anzubinden [5] [16].

2.2 Playbooks

Wie bereits im vorigen Abschnitt angeschnitten, bilden Playbooks das Zentrum für SOAR-Tools und generell für die Automatisierung von Prozessen. Entsprechend zentral sind Playbooks auch für diese Arbeit.

Die Notwendigkeit für Playbooks wird recht schnell deutlich, wenn man sich den Ablauf eines Vorfalls vorstellt. Angenommen, das CSIRT eines Unternehmens wird über eine Phishing-Mail im E-Mail-Postfach eines Mitarbeitenden informiert. Wie soll das CSIRT reagieren? Wird zunächst der Adressat der Mail für alle Mailkonten der Mitarbeitenden geblockt? Gibt es eine Möglichkeit, die Mail aus Postfächern der Mitarbeitenden automatisch zu löschen, falls sie an weitere Angestellte gesendet wurde? Sollte eine Warnmitteilung herausgegeben werden? Und in welcher Reihenfolge sollte all das passieren? Ein solcher Vorfall gehört bei größeren Unternehmen zur Tagesordnung. Es ist somit sinnvoll,

die resultierenden Prozesse zu standardisieren, was durch ein Playbook realisiert werden kann. Dadurch kann sichergestellt werden, dass im Falle eines Incidents alle notwendigen Schritte unternommen werden und dies nicht von der subjektiven Einschätzungen einzelner Personen im CSIRT abhängig wird. Zudem wird die Effizienz erhöht, da nicht zunächst darüber nachgedacht werden muss, was in welcher Reihenfolge zu tun ist.

Um die richtigen Reaktionen automatisiert ausführbar zu machen, benötigen Incident Response Tools eine klare Reihe von maschinenlesbaren Anweisungen, die darstellen, wie im Falle eines Incidents vorgegangen werden soll. Dies beschreibt nichts anderes als Playbooks, die die Automatisierung von Incident-Response-Prozessen erst ermöglichen [3].

2.2.1 Aufsetzen von Playbooks

Möchte eine Organisation ein Playbook erstellen, muss zunächst die Frage geklärt werden, ob das Playbook von Grund auf neu aufgesetzt werden soll, oder ob bestehende Playbooks für einen vergleichbaren Anwendungsfall zur Grundlage genommen und lediglich für den speziellen Use Case angepasst werden sollen. Viele Unternehmen, die SOAR-Tools anbieten, stellen hierfür beispielsweise bereits eine Reihe an vorgefertigten Playbooks zur Verfügung, die auch Community Playbooks genannt werden.

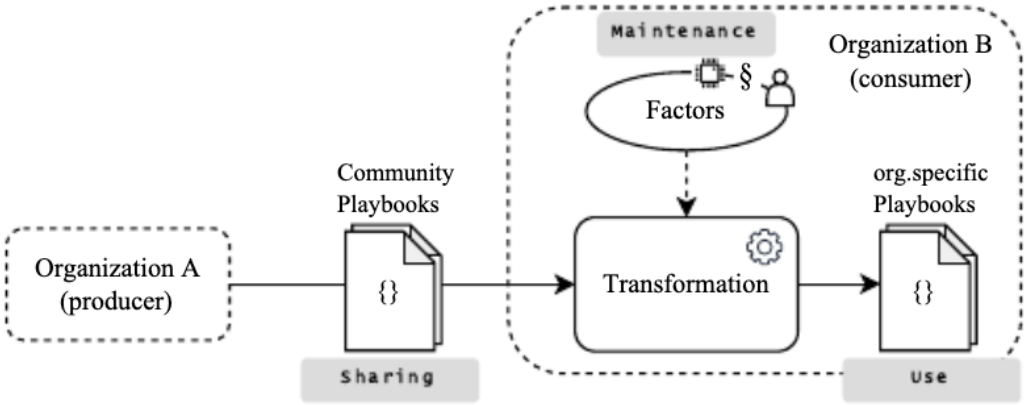

Abbildung 2.2: Nutzung von vorgefertigten Playbooks - aus [22]

In Abbildung 2.2 wird das Nutzen von Community Playbooks dargestellt. Eine Organisation A gibt eine Reihe an Playbooks heraus. Eine Organisation B, die ein Playbook aufsetzen möchte, wählt nun eins der bestehenden Playbooks aus. Diese sind absichtlich generisch gehalten. Entsprechend müssen zunächst die unternehmensspezifischen Faktoren geklärt werden. Vielleicht ist bei Organisation B die Netzwerkanalyse outgesourced oder es gibt Besonderheiten mit der Firewall. Das Community Playbook wird somit basierend auf diesen Faktoren transformiert und das organisationsspezifische Playbook entsteht [22].

2.2.2 Aufbau und Datenformate

Zwar geht aus den vorangehenden Abschnitten hervor, wieso Playbooks für Incident-Response-Prozesse - und gerade für die Automatisierung solcher - von entscheidender Bedeutung sind und wie Community Playbooks generell genutzt werden können, allerdings befinden sich diese Erläuterungen auf einer sehr abstrakten Ebene. Für die Praxis reicht es nicht einfach nur aus, zu wissen, dass es Playbooks braucht. Viel interessanter ist es, wie Playbooks tatsächlich aufgebaut sind - also was darin steht und wie die Daten vorliegen.

Diese Fragen stellen jedoch die Kernproblematik dar. Wie so oft gibt es auch bei Playbooks wenig Standardisierung. Selbst der Begriff des Incident-Response-Playbooks wird zwar versucht zu definieren, eine komplett einheitliche Definition gibt es allerdings nicht. Generell kann jedoch davon ausgegangen werden, dass Playbooks eine Beschreibung von Instruktionen beinhalten, die eine Hilfestellung für mehr oder weniger spezifische Angriffe, Bedrohungen oder Incidents aus dem Bereich der Cybersicherheit geben. Auch die möglichen Datenformate von Playbooks sind vielfältig. Der Grund hierfür ist aber auch ein rein praktischer. Die Frage, die sich bei jedem Playbook stellt, ist, für wen das Playbook primär gedacht ist. So könnte ein Playbook für einen Menschen textbasiert oder sogar grafisch aufgearbeitet sein, um mehr Übersicht zu gewährleisten. Für eine Maschine wäre es sinnvoller, eine code-basierte Darstellungsform zu wählen. Ein vereinfachtes Beispiel für ein Playbook könnte wie folgt aussehen:

Playbook Phishing Mail

- **Start** Eingang Phishing Mail
  - ✓ Ticketerstellung
  - ❏ E-Mail analysieren
  - ❏ E-Mail aus Posteingängen entfernen
  - ❏ User informieren und Sicherheitssysteme anpassen
- **End** Incident abgeschlossen

Auch eine solche Checkliste stellt bereits eine textbasierte Form eines Playbooks dar. In der Realität liegen Playbooks jedoch vorwiegend code-basiert vor. Eine simple Erklärung hierfür ist, dass diese so maschinenlesbar sind und somit auch für Menschen im Zweifelsfall durch Programme ansprechend und übersichtlich dargestellt werden können. Die Datenformate variieren hierbei. Bei einer Untersuchung von 1217 Playbooks von 14 verschiedenen Anbietern im Rahmen von [22] waren jedoch vorherrschend gängige Datenformate wie JSON, YAML und XML vertreten. Prozessorientierte Formate wie BPMN traten deutlich seltener auf.

Auch bei der Struktur und dem Aufbau dieser Community Playbooks gab es bei der Untersuchung aus [22] Gemeinsamkeiten. So stand immer im Kern des Playbooks der *Workflow* mit den einzelnen Steps. Diese Workflow Steps können ganz unterschiedlicher Natur sein und reichen von simplen HTTP-Requests bis hin zu komplexeren Entscheidungen, die manuell oder durch hinterlegte Bedingungen abgearbeitet werden.
Auch die zu übergebenden *Parameter* stehen im Zentrum der Playbooks. Die Informationen, welche Daten für den Input benötigt werden und welcher Output produziert wird, werden sowohl auf globaler Playbook-Ebene als auch für die einzelnen Steps bestimmt.
Weitere Komponenten sind *Meta-Informationen* wie Name, Beschreibung oder Versionierung und weiterführende *Features*, die hier zusammengefasst wurden als jegliche Features, die nur gelegentlich auftraten. Hierzu zählen beispielsweise Spezifikationen zum Deploymentprozess oder die Beschreibung von Testverfahren.

Zusammenfassend lässt sich also sagen, dass, auch wenn es je nach Anbieter diverse Unterschiede bei Aufbau und genutzten Datenformaten zwischen Playbooks gibt, dennoch

ein Großteil der Playbooks ähnliche Komponenten aufweisen und sich bei den Daten meist auf gängige Formate beschränkt wird.

2.3 Security Information Sharing

Wie bereits in der Einleitung erwähnt, verändert sich über die Jahrzehnte hinweg nicht nur die Internetlandschaft im Allgemeinen - auch die Cyber-Security-Landschaft muss sich anpassen an die stetige Professionalisierung der Cyberkriminalität. Durch diese Professionalisierung steigt das Risiko für Organisationen, dass bekannte Schwachstellen leicht ausgenutzt werden. Zusätzlich steigt auch die Skalierung und Komplexität von Systemen und Netzwerken, was die potenzielle Angriffsfläche noch weiter erhöht. Eine weitere Herausforderung hierbei ist, dass nach wie vor viele Aufgaben zur Erkennung von Angriffen innerhalb einzelner Organisationen durchgeführt werden und es nur wenig organisationsübergreifenden Informationsaustausch gibt. Der Informationsaustausch ist hier jedoch ein entscheidender Schritt, um ein gründliches Verständnis von groß angelegten Cyberangriffen zu erlangen. So stellt dieser Austausch eine der Schlüsselkonmponenten für den Schutz von zukünftigen Netzwerken und Systemen dar. Die möglichen Anwendungsfälle reichen hier von der Entdeckung versteckter Cyberangriffe und Malware über Frühwarnungen bis hin zu Ratschlägen zur Sicherung von Netzwerken und der gezielten Verbreitung von Threat Intelligence Data [25].

2.3.1 MISP

Die prominenteste Plattform hierfür stellt MISP dar. Begonnen wurde mit der Entwicklung bereits im Jahre 2011. Christophe Vandeplas war damals so frustriert vom Stand der Dinge, IOCs (Indicators of Compromise) über E-Mail oder PDFs auszutauschen, statt ein maschinenlesbares Format zu verwenden, dass er die Entwicklung einer Alternativmöglichkeit in Angriff nahm. Nur einen Monat nachdem er seine Implementierung bei seinem Arbeitgeber - dem belgischen Verteidigungsministerium - vorstellte, wurde das Projekt offiziell auch dort eingeführt. Nach kurzer Zeit stieg auch die NATO in das Projekt ein und der Name Malware Information Sharing Platform (MISP) wurde gewählt [20].

Trotz dieses Namens bietet MISP heutzutage weitreichendere Funktionalitäten an. Es geht nicht mehr nur um Malware Indikatoren - mittlerweile wurde dies um Betrugs-

und Schwachstelleninformationen erweitert. Hierin begründet sich die Kernfunktionalität von MISP: eine geteilte Datenbank, die es ermöglicht, technische und nicht-technische Informationen zu speichern. Weitere Funktionalitäten werden in der nachfolgenden Liste beschrieben. Hierbei gilt zu beachten, dass dies nur einen Teil der Funktionalitäten aufführt und diese nicht umfassend beschreiben soll. Die Liste basiert hierbei auf der offiziellen Website von MISP, wo weitergehende Beschreibungen gefunden werden können [19].

Funktionalitäten

1. Automatische Korrelation zur Ermittlung von Beziehungen zwischen Attributen und Indikatoren aus Malware, Angriffskampagnen oder Analysen
2. Ein flexibles Datenmodell, in dem komplexe Objekte ausgedrückt und miteinander verknüpft werden können, um Bedrohungsdaten, Vorfälle oder verbundene Elemente darzustellen
3. Eingebaute Sharing-Funktionalität zur Vereinfachung des Datenaustauschs unter Verwendung verschiedener Verteilungsmodelle
4. Erweiterte Filterfunktionen und Warnlisten, die den Analyst*innen helfen, Ereignisse und Attribute zu melden
5. Eine intuitive Benutzeroberfläche für User zum Erstellen, Aktualisieren und gemeinsamen Bearbeiten von Ereignissen und Indikatoren
6. Speicherung von Daten in einem strukturierten Format (was eine automatisierte Nutzung der Datenbank für verschiedene Zwecke ermöglicht) mit umfassender Unterstützung von Cybersicherheits- und Betrugsindikatoren
7. Eine Vielzahl an unterstützten Datenformaten für Import und Export

2.3.2 Automatische Warnmeldungen

Eine Möglichkeit, die Daten von MISP indirekt zu nutzen, stellt der AW-Service des DFN-CERTs dar. AW steht hier für 'Automatische Warnmeldungen'. Die Grundidee dieses AW-Services ist in Abbildung 2.3 zu sehen und kann folgendermaßen erläutert werden:

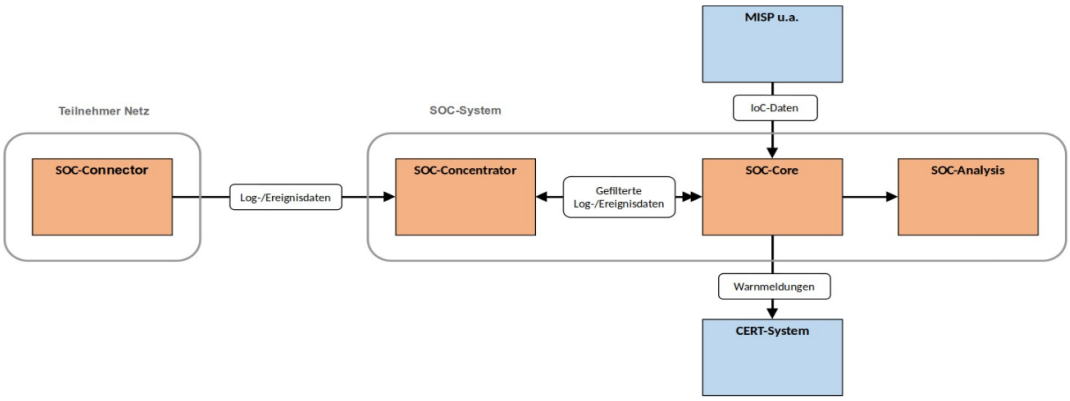

Abbildung 2.3: Analyseprozess für AW-Meldungen - aus [6]

Eine Organisation installiert den sogenannten SOC-Connector in ihrem System. Dieser dient der Einlieferung von Logdaten in das Analysesystem des DFN-CERTs. Konkret werden diese zunächst an ein Concentrator-Cluster übertragen. Das Cluster nimmt eine erste Filterung, Normalisierung und Analyse der Daten vor, bevor sie im Anschluss an die zentrale Core-Komponente weitergeleitet und analysiert werden. Durch die von MISP bereitgestellten IoC-Daten können die bestehenden Daten gegebenfalls weiter angereichert werden. Die Analyse ergibt schließlich, ob es sich um einen sicherheitsrelevanten Vorfall handelt. In diesem Fall wird daraufhin eine AW-Meldung erzeugt [6] [18].

An der HAW Hamburg wird der beschriebene AW-Service aktiv genutzt, um Informationen zu Bedrohungen und IoCs zu erhalten. Diese AW-Meldungen sind zentral für die Automatisierungsstrategie, die in dieser Arbeit behandelt wird. Die AW-Meldungen der vergangenen Monate stellen die Datensätze dar, die im Rahmen dieser Arbeit bearbeitet wurden. Wie diese Warnmeldungen konkret aussehen und welche Erkenntnisse hieraus gewonnen werden können, wird im nachfolgenden Kapitel genauer betrachtet.

3 Daten

Das folgende Kapitel bildet die Grundlage für diese Arbeit: die Daten. Für diese Arbeit wurden alle AW-Meldungen betrachtet, die die HAW Hamburg zwischen dem 27.01.2024 und dem 10.09.2024 vom DFN-CERT erhalten hat. Für die Bearbeitung dieser Daten wurde bereits in [15] ein Grundstein gelegt, wo unter anderem erste Möglichkeiten erwogen wurden, wie diese Daten gefiltert werden könnten. Die vorliegende Arbeit erweitert diese Ansätze. Nach einem kurzen Überblick über den bisherigen Stand der Daten aus [15] wird ein noch genauerer Blick auf die einzelnen Datensätze geworfen. Darauf basierend werden Filtermöglichkeiten erarbeitet und vorgestellt. Zuletzt wird beschrieben, welche Daten überhaupt übrig bleiben.

3.1 Stand der Dinge

Da diese Arbeit an vorherige Arbeiten anknüpft, wird im folgenden Abschnitt der Stand der Dinge vor dieser Arbeit beschrieben. Dazu zählt insbesondere, welche Daten nun vorliegen und wie diese Daten bereits bearbeitet wurden.

3.1.1 Datenformat

In [15] wurden die Daten nicht nur gefiltert, sondern zunächst aufbereitet. Diese Aufbereitung umfasste die Überführung der Daten in ein gültiges CSV-Format. Anschließend wurden zwei Skripte zur Anonymisierung und zur Deanonymisierung verfasst. Dies war insbesondere notwendig, um die Daten automatisiert bearbeiten und sie im Rahmen dieser Arbeit auch veröffentlichen zu können. Die Skripte anonymisieren, indem bestimmte Teile wie IP-Adressen oder URLs neue Werte zugewiesen bekommen. Dieses Mapping wird dann in einer JSON-Datei gespeichert, die im Anschluss wiederum dazu genutzt werden kann, die anonymisierten Daten zu deanonymisieren.

In dieser Arbeit werden ausschließlich die anonymisierten Daten besprochen. In Abbildung 3.1 kann ein Beispiel für einen solchen Datensatz betrachtet werden. Für diese Arbeit irrelevant sind die Spalten Netzsegment/Domain und die Meldungs-ID. Erstere trägt fast immer den gleichen Wert und Letztere dient nur der Identifizierung.

Datum	Kategorie	Netzsegment/Domain	Betroffene Ressource	Ziel	Zusätzliche Informationen	Meldungs-ID
2024-01-29 00:04	Bot	Hochschule	127.0.0.1:87	127.0.0.2:81	CERT DIAGNOSIS:bisar, OCCURENCES:1	2401-243-1081

Abbildung 3.1: Beispiel einer AW-Meldung, aus [15]

Ein wichtiger Faktor zur Einteilung der AW-Meldungen ist die Spalte *Kategorie*. Hier wird der Meldungstyp der jeweiligen Warnmeldung aufgeführt. Ein näherer Blick auf die Meldungstypen wird in [15] geworfen, für diese Arbeit sind insbesondere die folgenden relevant:

- **Bot:** Dieser Meldungstyp dient als Indikator dafür, dass eine betroffene Ressource womöglich Teil eines Botnets ist.
- **Configuration:** Hierdurch wird darauf hingewiesen, dass bestimmte Konfigurationen des Systems zu Sicherheitsrisiken führen können.
- **Vulnerability:** Dies deutet auf eine konkrete Schwachstelle im System hin, gegen die die betroffene Ressource nicht ausreichend geschützt ist und entsprechend ausgenutzt werden kann.
- **Security Monitoring/Blocklist:** Eine solche Warnmeldung gibt an, dass eine betroffene Ressource in einer öffentlichen Blockliste geführt ist.

Weitere entscheidende Faktoren für die Analyse sind die *Betroffene Ressource* und das *Ziel*. Wie der Name schon vermuten lässt, ist Ersteres die betroffene Ressource, die hier durch IP-Adresse und Port identifiziert werden kann. Auch das von der betroffenen Ressource kontaktierte Ziel wird durch IP-Adresse und Port beschrieben. Für beide Felder können jedoch auch lediglich eine der beiden Informationen (IP oder Port) oder keine vorhanden sein. Die Spalte *Zusätzliche Informationen* kann weitere Hinweise für die Warnmeldung liefern. Hierzu zählen beispielsweise der Name der Malware oder eine Target URL (URL auf dem kontaktierten System).

3.1.2 Überblick über bisherige Filter

In [15] wurden für die vorliegenden Daten vier der darin beschriebenen Filter realisiert. Dies führte bereits zu einer Reduzierung der Datensätze auf fast ein Achtel der ursprünglichen Einträge.

Eine kurze Beschreibung der bisherigen Filter:

- **Doppelte Meldungen:** Zwei der Hauptdatenquellen des DFN-CERTs für die Warnmeldungen sind das BSI (Bundesamt für Sicherheit in der Informationstechnik) und Shadowserver. Shadowserver ist eine gemeinnützige Organisation, die Daten über diverse sicherheitsrelevante Bereiche wie Malware oder Botnetze sammelt [23]. Da das BSI ebenfalls auf Shadowserver als Ressource zugreift, kann es dazu kommen, dass durch indirekte und direkte Meldungen beim DFN-CERT doppelte Meldungen ausgesendet werden.
- **Bekannte Dienste:** Ein Teil der Warnmeldungen geht auf bekannte Dienste innerhalb des Systems zurück. Wenn beispielsweise innerhalb des Systems ein Portscan-Dienst läuft, können je nach kontaktierten Zielen durch diese Scans Warnmeldungen ausgelöst werden. Entsprechend sind diese Einträge aber für den Incident-Handling-Prozess der Hochschule nicht relevant.
- **Hohe Ports:** Da hohe Ports von Systemen generell für temporäre oder weniger sicherheitskritische Dienste vergeben werden, sind Warnmeldungen für entsprechende betroffene Ressourcen häufig ebenfalls weniger relevant. Hierzu zählen insbesondere Portnummern, die höher als 1.024 sind. Diese Meldungen werden zwar nicht gänzlich herausgefiltert, aber werden mit einer niedrigeren Priorität versehen.
- **Schwachstellen aus Inventarscans:** Da das DFN-CERT regelmäßig Inventarscans durchführt, die lediglich potenzielle Schwachstellen in laufenden Systemen ohne direkt Bedrohung melden, sind diese Meldungen nicht unmittelbar relevant für den Incident-Handling-Prozess. Stattdessen müssen diese im Rahmen des Schwachstellenmanagements separat verarbeitet werden, was wiederum zwar in bestimmten Fällen eine Teilaufgabe von CSIRTs ist (siehe 2.1.3), aber da es sich bei dieser Arbeit um Filter für die unmittelbare Incident Response dreht, nur einen weiteren Prozess anstößt.

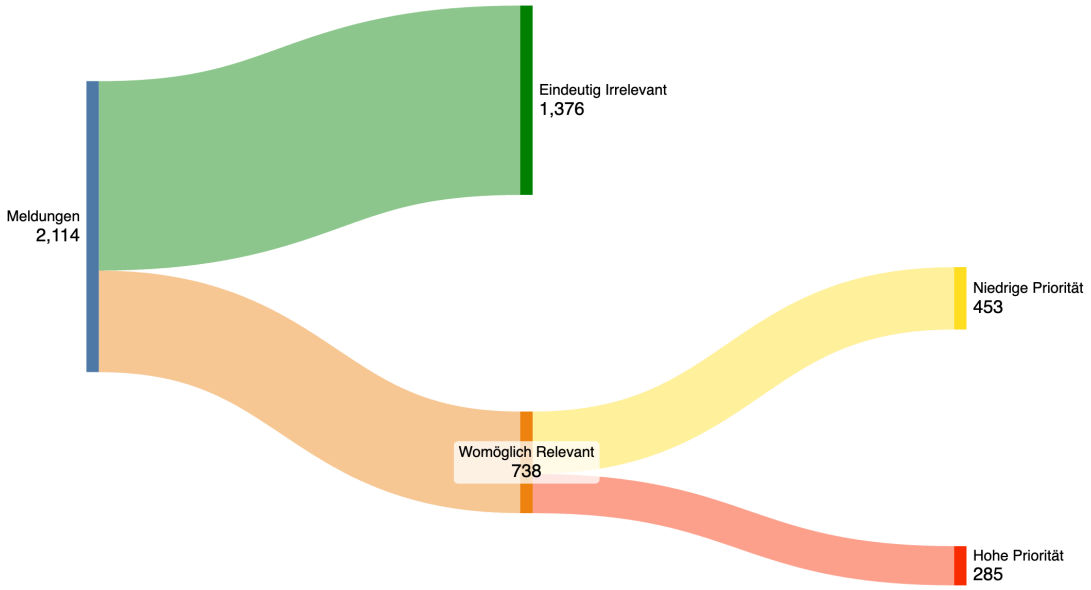

Abbildung 3.2: Sankey-Diagramm der Filterwellen, aus [15]

Wie in Abbildung 3.2 veranschaulicht wird, wurden die Filter bisher nach *Eindeutig Irrelevant* und *Womöglich Relevant* eingeteilt. *Doppelte Meldungen* und *Bekannte Dienste* zählen zu den Filtern für eindeutig irrelevant, während die anderen beiden Filter für womöglich relevant stehen. Die Idee ist hierbei, dass jede Warnmeldung zunächst durch die eindeutigen Filter geht. Wird die Meldung herausgefiltert, kann sie direkt archiviert werden. Andernfalls wird die Priorität der Warnmeldung dadurch bestimmt, ob einer der übrigen Filter zutrifft. Ist das der Fall, handelt es sich um eine Meldung mit niedriger Priorität. Sonst wird ein Dringlichkeitsvermerk für die Warnmeldung beigefügt.

Im Grunde wird dieser Prozess auch bei der vorliegenden Arbeit beibehalten. Im nächsten Teil dieses Kapitels wird jedoch beschrieben, wie die eindeutigen Filter noch weiter unterteilt werden.

3.2 Weitere Datenanalyse

Die Daten, die nach der Anwendung der Filter aus [15] übrig bleiben, sind zwar schon deutlich reduziert, der folgende Abschnitt untersucht diese Daten allerdings noch weiter und stellt Möglichkeiten vor, wie darauf basierend weitere Filter konzipiert werden können, um die relevanten Warnmeldungen weiter einzugrenzen.

3.2.1 Ein genauer Blick auf die Daten

Um die Daten weiter zu filtern, müssen die übrigen Datensätze genauer betrachtet werden. Eine Möglichkeit hierfür, die in [15] nicht genutzt wurde, ist, die Meldungen basierend auf ihrem Datum zu analysieren. Häufungen können hier Untermengen ergeben, die weitere Filteroptionen mit sich ziehen.

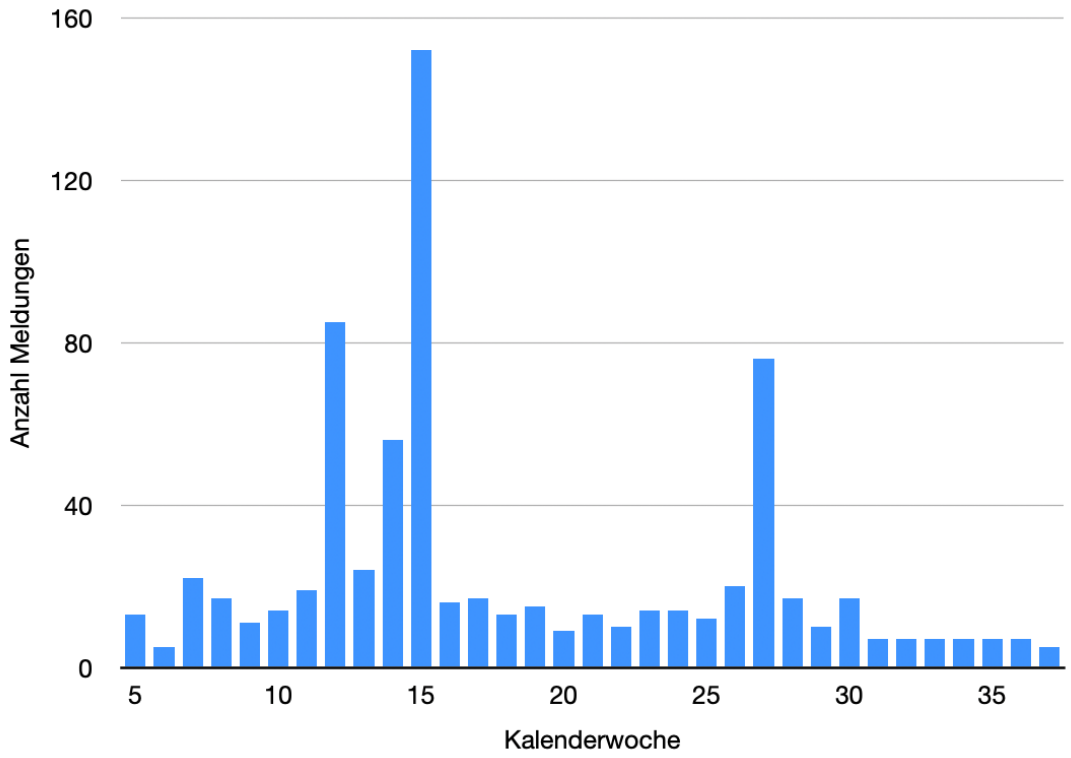

Abbildung 3.3: Zeitstrahl der gefilterten Meldungen

Abbildung 3.3 zeigt hierzu einen Zeitstrahl der Meldungen, die basierend auf den bisherigen Filtern als *womöglich relevant* klassifiziert wurden. Es fallen deutlich die höheren Eingänge an Meldungen in den Kalenderwochen 12-15 und 27 auf. Hierzu lassen sich allerlei Mutmaßungen anstellen. Beispielsweise handelt es sich hierbei an einer Hochschule um besonders prüfungsrelevante Zeiträume. Insofern könnte es sein, dass bestimmte Systeme von Studierenden oder Lehrpersonal überproportional frequentiert werden. Dies ist aber nur eine von vielen möglichen Erklärungen.

Für diese Arbeit besonders interessant ist, welche Meldungen in diesen Zeiträumen eingehen. Schaut man sich beispielsweise die Daten in Kalenderwoche 15 an, fällt ziemlich

schnell eine IP-Adresse ins Auge, die als betroffene Ressource für 140 der 152 Warnmeldungen in dieser Woche verantwortlich ist. Wie schon bei dem bekannten Portscan-Dienst handelt es sich hier bei allen Warnmeldungen um die Kategorie Bot (manchmal mit der Unterkategorie Bot/HTTP versehen). Ein weiterer Hinweis liefert ein Blick auf die Ziel-Ports. Bei allen Meldungen in dieser Zeit, die als Ressource diese eine IP-Adresse haben, gibt es nur zwei verschiedene Portnummern, die beim Ziel auftreten. Der Filter *Bekannte Dienste* umfasst bereits einen Portscan-Dienst, der ein ähnliches Verhalten aufweist, da auch bei diesem immer ein spezifischer Port als Ziel-Port auftritt. Dies legte die Vermutung nahe, dass es sich in diesem Fall ebenfalls um einen Portscan-Dienst handeln könnte. Nach Rücksprache mit den zuständigen Verantwortlichen konnte diese Vermutung verifiziert werden.

Die Häufung um Kalenderwoche 27 basiert auf einer Reihe an gleichen Vulnerability-Warnmeldungen für unterschiedliche betroffene Ressourcen. Vermutlich wurde hier somit vor einer Schwachstelle auf unterschiedlichen Systemen gewarnt, was sinnvolle Warnmeldungen beschreibt und somit keine direkte Filtermöglichkeit darbietet. Da es sich um die gleiche Schwachstelle handelt, könnte es sinnvoll sein, Referenzen zu bereits bearbeiteten Tickets zu erstellen, um die Vorgehensweise, die für ein System funktioniert hat, auch bei den restlichen anzuwenden. Die Warnmeldungen deshalb immer herauszufiltern, ergibt jedoch nicht unbedingt einen Sinn, da die Schwachstelle in jedem Fall bei allen Systemen gefixed werden muss.

Möglicherweise könnten mit den Daten aus mehreren Jahren noch weitere Erkenntnisse gezogen werden, um saisonale Häufungen stärker zu beleuchten. Die Datensätze für ein einzelnes Jahr reichen hier jedoch kaum aus.

Eine weitere Möglichkeit, die Warnmeldungen nach den zugrundeliegenden Wochentagen zu gruppieren, wurde in Abbildung 3.4 versucht. Wie sich zeigt, gibt es tatsächlich leichte Schwankungen - im Schnitt gibt es so an einem Freitag nur etwa halb so viele Meldungen wie an einem Mittwoch. Blickt man allerdings auf die tatsächlichen Daten, ergab sich in der Analyse dieser Arbeit kein klarer Grund hierfür. Somit konnten keine weiteren Erkenntnisse hieraus gezogen werden, die eine Optimierung, wie in der Form von neuen Filteroptionen, nahelegen würden.

Doppelte Meldungen neu gedacht Im vorigen Abschnitt wurde bereits der bestehende Filter *Doppelte Meldungen* vorgestellt. Hierbei handelt es sich um fehlerhaft doppelte Meldungen, die vom DFN-CERT gesendet werden, da das DFN-CERT aus

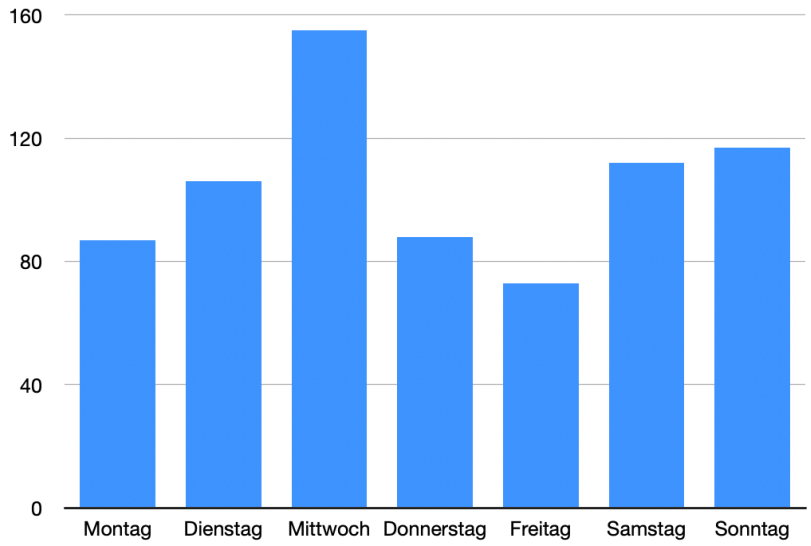

Abbildung 3.4: Meldungsanzahl nach Wochentagen

mehreren Quellen seine Daten bezieht, die sich mitunter überschneiden. Bei der weiteren Datenanalyse, die in dieser Arbeit getätigt wurde, ist allerdings noch ein weiteres Beispiel für eine Art 'doppelte Meldungen' aufgefallen. Man betrachte folgendes Beispiel:

Das DFN-CERT meldet im April eine Schwachstelle für eine betroffene Ressource. Nun vergeht einige Zeit und die Warnmeldung für diese Schwachstelle geht im Incident-Handling-Prozess der HAW Hamburg unter. Plötzlich ist es August und die Schwachstelle wurde nicht gefixed. In diesem Fall bleibt das DFN-CERT eventuell nicht stumm. Stattdessen schickt es immer wieder eine AW-Meldung bezüglich der bekannten Schwachstelle. Es handelt sich hierbei um wiederholte Warnmeldungen, denen dasselbe Problem zugrundeliegt.

Hieraus stellt sich die Frage, wie mit derartigen Warnmeldungen umgegangen werden sollte. Eine Option wäre es, die Warnmeldungen einfach direkt herauszufiltern. Der Incident-Handling-Prozess für diese Schwachstelle wurde bereits eingeleitet. Wenn also in dieser Zeit 20 gleiche Warnmeldungen eingehen, wäre es fragwürdig, 20 verschiedene Prozesse neu zu starten. Immer einen komplett neuen Prozess zu erstellen, würde im Zweifelsfall also keinen Sinn ergeben. Nun ist es aber bereits August und die Schwachstelle wurde in vier Monaten nicht behoben. Die Option, einfach alle Warnmeldungen herauszufiltern, hat somit ebenfalls einen Nachteil. Die stetigen Erinnerungen, die die vielen Warnmeldungen in der Zwischenzeit dargestellt hätten, wurden einfach ignoriert, da es

schlichtweg keine Erinnerungen gab. Ein Zwischenweg könnte jedoch die Lösung bringen. Es könnte einen Zeitraum geben, der als gerade noch angemessen für die Bearbeitung von Schwachstellen gelten könnte - beispielsweise 90 Tage. Gehen in diesen 90 Tagen wiederholte Meldungen für die gleichen Schwachstellen ein, werden diese herausgefiltert. Ist die ursprüngliche Meldung allerdings älter als 90 Tage, wird eine interne Prüfung veranlasst, wieso die Schwachstelle in dieser Zeit nicht behoben wurde, um zu verhindern, dass Schwachstellenmeldungen einfach untergehen können.

Datum	Kategorie	Netzsegment/ Domain	Betroffene Ressource	Ziel	Zusätzliche Informationen	Meldungs-ID	CERT DIAGNOSIS	Kalenderwoche	Wochentag
2024-06-24 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt ()	2406-541-1150		26	1
2024-06-25 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt ()	2406-541-1150		26	2
2024-06-26 09:01:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt ()	2406-375-7246		26	3
2024-06-27 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt ()	2406-866-3129		26	4
2024-06-28 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt ()	2407-742-5493		26	5
2024-06-29 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt ()	2407-742-5493		26	6
2024-06-30 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt ()	2407-742-5493		27	7
2024-07-01 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt (dnsrd)	2407-580-7494		27	1
2024-07-02 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt (dnsrd)	2407-474-0940		27	2
2024-07-03 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt (dnsrd)	2407-541-7769		27	3
2024-07-04 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt (dnsrd)	2407-822-7427		27	4
2024-07-05 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt (dnsrd)	2407-704-7071		27	5
2024-07-06 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt (dnsrd)	2407-704-7071		27	6
2024-07-07 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt (dnsrd)	2407-704-7071		28	7
2024-07-08 09:02:00	Security Monitoring/ Blocklist	Hochschule	127.0.0.1		RECORD:Dataplane.org - DNS Recursion Attempt (dnsrd)	2407-203-2472		28	1

Abbildung 3.5: Anonymisierter Ausschnitt der Security Monitoring/Blocklist Wiederholungsmeldungen

Dieses Phänomen der Wiederholungsmeldungen tritt wohlgemerkt nicht nur bei Warnmeldungen des Meldungstyps Vulnerability auf. Ein Blick auf die Daten zeigt beispielsweise, dass an jedem einzelnen Tag ab dem 24. Juni bis zum Ende der Datensätze gegen 9 Uhr morgens eine bestimmte Warnmeldung einging. Ein Ausschnitt dieser Daten kann in Abbildung 3.5 betrachtet werden. Hierbei handelt es sich um insgesamt 79 gleiche Meldungen des Typs Security Monitoring/Blocklist und damit um alle Warnmeldungen dieses Meldungstyps in den vorliegenden Datensätzen. Dieser Meldungstyp gibt Hinweis darauf, dass die betroffene Ressource auf einer öffentlichen Blocklist aufgeführt ist. Die zugrundeliegende IP-Adresse, die sich hierhinter verbirgt, ist keine andere, als die IP-Adresse, auf dem einer der bekannten Portscan-Dienste aktiv ist. Dies erklärt die Listung dieser IP-Adresse. Es gibt diverse Möglichkeiten, die als Lösung dieses Problems in Betracht gezogen werden können, die in Kapitel 4 näher beleuchtet werden. Fest steht allerdings, dass

diese Warnmeldungen, wie schon bei den oben besprochenen Schwachstellenmeldungen, reine Wiederholungen darstellen und ähnlich behandelt werden sollten.

Ähnliches lässt sich bei den AW-Meldungen der Kategorie *Configuration* verzeichnen. Auch hier lassen sich eine Reihe an Wiederholungsmeldungen in den Datensätzen finden. Sowohl bei den Warnmeldungen des Typs *Security Monitoring/Blocklist* als auch bei denen des Typs *Configuration* verhält es sich allerdings anders als bei Warnmeldungen zu Schwachstellen. Bei einer Schwachstelle gibt es eine klare Handlungsanweisung: die Schwachstelle zu beheben. Bei einer Konfiguration kann es möglicherweise eine abgewogene Entscheidung sein, die den Aufwand oder einen möglichen Nutzen gegen das Risiko aufwiegt. Auch bei den Warnmeldungen zu Einträgen in einer Blocklist kann es Gründe geben, dass die Warnmeldungen weiter eingehen, weil beispielsweise ein Entfernen von der Blocklist nicht erfolgreich war. Entsprechend kann in solchen Fällen eine lokale Whitelist in Betracht gezogen werden, um die weitere Bearbeitung von gleichen Warnmeldungen, mit denen sich bereits befasst wurde, zu unterbinden.

3.2.2 Neue Filtermöglichkeiten

Aus dieser weiteren Datenanalyse ergeben sich zum einen einige neue Filteroptionen und zum anderen können bestehende Filter weiter angepasst werden:

- **Bekannte Dienste:** Zunächst kann der bereits existierende Filter für bekannte Dienste um den weiteren, gefundenen Portscan-Dienst erweitert werden.
- **Wiederholungsmeldungen:** Eine neue Filtermöglichkeit basiert auf der Erkenntnis der wiederholten Warnmeldungen. Dies gilt insbesondere für die folgenden Meldungstypen:
  - *Vulnerability:* Gehen wiederholte Warnmeldungen für die gleiche Schwachstelle bei der selben betroffenen Ressource ein, werden diese ignoriert, sollte die ursprüngliche Meldung kürzer als 90 Tage zurückliegen.
  - *Security Monitoring/Blocklist:* Auch bei diesem Meldungstyp werden die Warnmeldungen ignoriert, wenn die Ursprungsmeldung jünger als 90 Tage ist. Hier gibt es allerdings auch die weitere Option der Whitelist, die auch eine Filterung über die 90 Tage heraus ermöglicht.

- *Configuration:* Hierbei wird der gleiche Prozess wie beim Meldungstyp 'Security Monitoring/Blocklist' durchlaufen. Es gibt sowohl die Möglichkeit wiederholte Meldungen herauszufiltern als auch durch eine Whitelist eine weitere Filteroption einzuführen.

3.2.3 Filterkategorien

Bevor die Ergebnisse der neuen und bestehenden Filteroptionen vorgestellt werden, sollte zunächst ein Blick auf die Kategorisierung der Filter geworfen werden. Wie oben in Abbildung 3.2 zu sehen ist, wurde in [15] noch eine Einteilung der Filter in zwei Wellen vorgenommen. Die erste Filterwelle filterte die Daten in *Eindeutig Irrelevant* und *Womöglich Relevant*, bevor die zweite Welle an Filtern die Daten weitergehend anhand ihrer Priorität bewertete.

Würde man die Filter nach den bisherigen Kategorien einteilen, sehe das Ergebnis folgendermaßen aus:

- **Relevanzbewertung:**
  - Doppelte Meldungen
  - Bekannte Dienste
  - Wiederholungsmeldungen
- **Prioritätsbewertung:**
  - Hohe Portnummern
  - Schwachstellen aus Inventarscans

In dieser Arbeit bietet sich nach den neuen Filtermöglichkeiten auch eine neue und feingranularere Kategorisierung der Filter an. Dies liegt primär daran, dass sich die drei Filteroptionen für die Relevanzbewertung elementar unterscheiden. Die doppelten Meldungen stellen tatsächliche Fehlmeldungen dar. Der einzige Grund, wieso diese eingehen, ist, dass das DFN-CERT die selben Meldungen von unterschiedlichen Quellen doppelt meldet. Die Meldungen der Filteroption *Bekannte Dienste* stellen hingegen korrekte Meldungen dar, die schlichtweg für den Incident-Handling-Prozess der HAW Hamburg nicht weiter relevant sind, da die zugrundeliegenden Dienste kein Risiko darstellen. Wiederum

anders verhält es sich mit den Wiederholungsmeldungen. Hierbei handelt es sich zum einen um korrekte Meldungen. Zum anderen geht von ihnen aber auch potenziell ein Risiko aus - der einzige Grund, wieso diese nicht für das restliche Incident Handling in Frage kommen, ist, weil die Ursache bereits in Bearbeitung ist.

Es folgt eine logische Trennung der entsprechenden Filterkategorien, die in Zukunft potenziell um weitere Filter ergänzt werden können:

- **Fehlmeldungen:**
  - Doppelte Meldungen
- **Risikobewertung:**
  - Bekannte Dienste
- **Relevanzbewertung:**
  - Wiederholungsmeldungen
- **Prioritätsbewertung:**
  - Hohe Portnummern
  - Schwachstellen aus Inventarscans

3.3 Gefilterte Daten

Aus den neuen Filtermöglichkeiten ergibt sich die Frage, welche Daten denn tatsächlich übrig bleiben, wenn die bereits existierenden sowie die neuen Filteroptionen ausgeschöpft werden. Diese resultierenden Daten werden im folgenden Abschnitt betrachtet.

3.3.1 Filtereffektivitäten

Ein erster Schritt ist hier der Blick auf die verschiedenen Effektivitäten der einzelnen Filter zu werfen. Diese sind in Abbildung 3.6 visualisiert. Besonders anzumerken ist hier der Anstieg auf 1.637 gefilterte Warnmeldungen durch die Filteroption *Bekannte Dienste*. In [15] lag diese Zahl noch bei 1.195. Auch durch den neuen Filter für Wiederholungsmeldungen kann eine große Anzahl an Warnmeldungen herausgefiltert werden. Dies beträgt einen Anteil an 849 Einträgen.

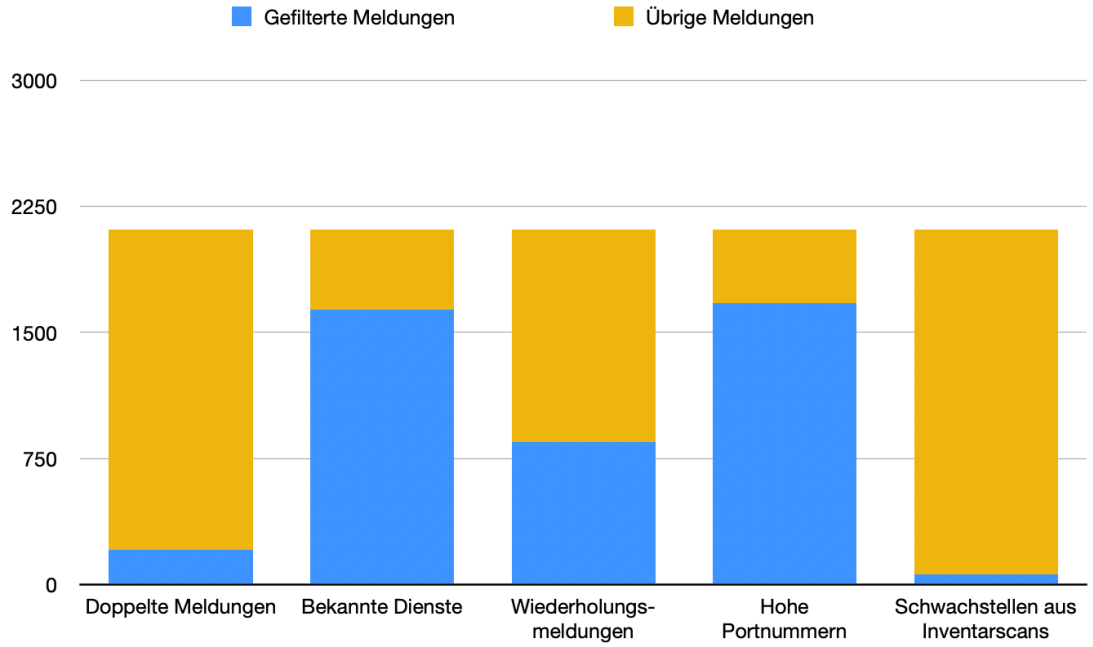

Abbildung 3.6: Anzahl der herausgefilterten Meldungen pro Filter (alleinstehend ausgeführt)

Mit einem Blick auf Abbildung 3.6 fällt jedoch auch auf, dass sich die Filter in den herausgefilterten Daten überschneiden müssen, da die Anzahl der gefilterten Daten die Anzahl der ursprünglichen Einträge übersteigt. Besonders interessant ist also vielmehr, was passiert, wenn die Filter sequenziell ausgeführt werden und somit wie viele Einträge tatsächlich nach der Anwendung aller Filter übrig bleiben.

Die sequenzielle Ausführung der Filter liefert ein signifikantes Ergebnis, was in Abbildung 3.7 betrachtet werden kann. Die Warnmeldungen, die tatsächlich im weiteren Incident Handling bearbeitet werden müssen, sind unter *Neue Meldungen* zu finden. Mit nur noch

92 von ursprünglich 2.114 Warnmeldungen liegt die Anzahl bei etwa 4% der ursprünglichen Werte. Somit muss im Schnitt nur noch jede 25. Warnmeldung tatsächlich bearbeitet werden. Des Weiteren haben von diesen 92 Meldungen nur 11 eine hohe Priorität. Da die Daten über einen Zeitraum von etwa 8 Monaten gesammelt wurden, bedeutet das nur etwas mehr als eine hoch priorisierte Warnmeldung pro Monat.

Es bietet sich an, an dieser Stelle einen Vergleich zu ziehen mit der Arbeit in [15]. Während zuvor noch 738 Einträge in den weiteren Incident-Handling-Prozess übergegangen wären, wurde diese Zahl mit nunmehr 92 massiv reduziert. Obwohl die tatsächliche Anzahl der täglich manuell zu bearbeitenden Warnmeldungen starken Tagesschwankungen unterliegt, hilft es, dieses statistische Mittel zu betrachten, um den großen Unterschied zu veranschaulichen, der sich an dieser Stelle auftut. Bei einem Zeitraum von 228 Tagen bedeutet die Reduzierung von 738 auf 92 Einträgen, dass statt etwas über 3 Warnmeldungen pro Tag nun im Schnitt seltener als an jedem zweiten Tag eine Warnmeldung manuell bearbeitet werden muss.

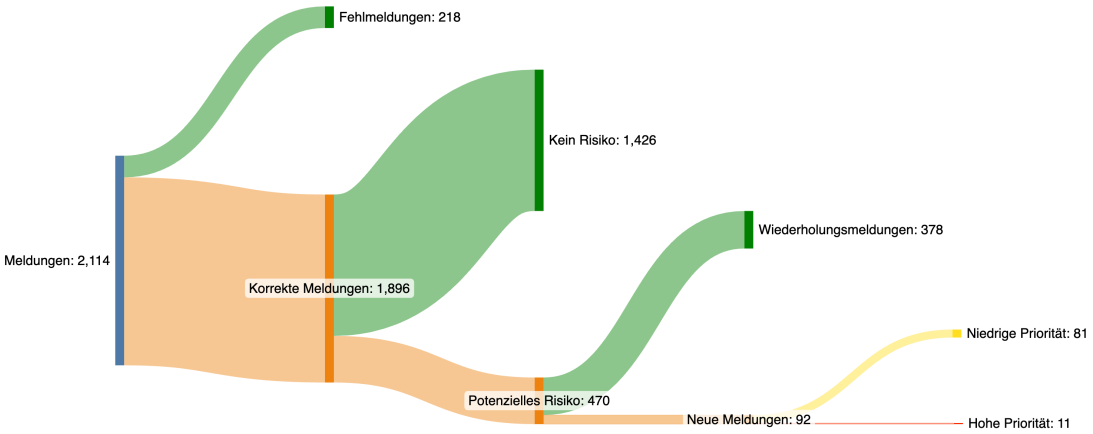

Abbildung 3.7: Sequenzielle Anwendung der Filterkategorien

3.3.2 Resultierende Daten

Nachdem nun die Effektivität der Filter unter Beweis gestellt wurde, bleibt noch die Frage zu klären, wie die Daten aussehen, die tatsächlich im Incident Handling bearbeitet werden müssen. Abbildung 3.8 stellt hierzu eine klare Verteilung dar. Für Einträge auf öffentlichen Blocklisten gab es tatsächlich nur einen Fall in den Datensätzen. Ähnlich gering sieht es bei dem Meldungstyp *Configuration* aus, bei der ebenfalls nur 4 Warnmeldungen zu registrieren sind. Bei dem Rest der 92 AW-Meldungen halten sich Bot

und Vulnerability in etwa die Waage. Sieht man sich die zugrundeliegenden Daten an, fallen allerdings auch hier noch weitere Muster auf. So sind beispielsweise 39 der 44 Vulnerability-Meldungen an zwei Tagen eingegangen und gleichen einander stark. Dabei handelt es sich bei allen um die gleiche Schwachstelle, die bei unterschiedlichen betroffenen Ressourcen verzeichnet wurde. Diese Meldungen wurden als Teil des Inventarscans verschickt, der einen eigenen Filter besitzt und hier könnte je nach Strukturierung der Organisation eine rasche Bearbeitung resultieren, da sich die Behebung der Schwachstellen als verhältnismäßig repetitiv herausstellen sollte. Wie oben beschrieben, wurde sich dennoch dazu entschieden, diese Warnmeldungen nicht weiter herauszufiltern, da es sich um tatsächlich relevante Warnmeldungen handelt und sie somit dem Incident Handling übergeben werden sollten. Für die Zukunft könnte es dennoch von Vorteil sein, auch hierfür automatisierte Prozesse zu entwickeln, um vergleichbar ähnliche Aufgaben zu sammeln. Besonders wenn es zentralisierte Stellen zur Behebung von Schwachstellen gibt, die über die Grenzen von einzelnen Untersystemen hinaus verantwortlich sind, könnte eine Aggregation dieser Meldungen sinnvoll sein.

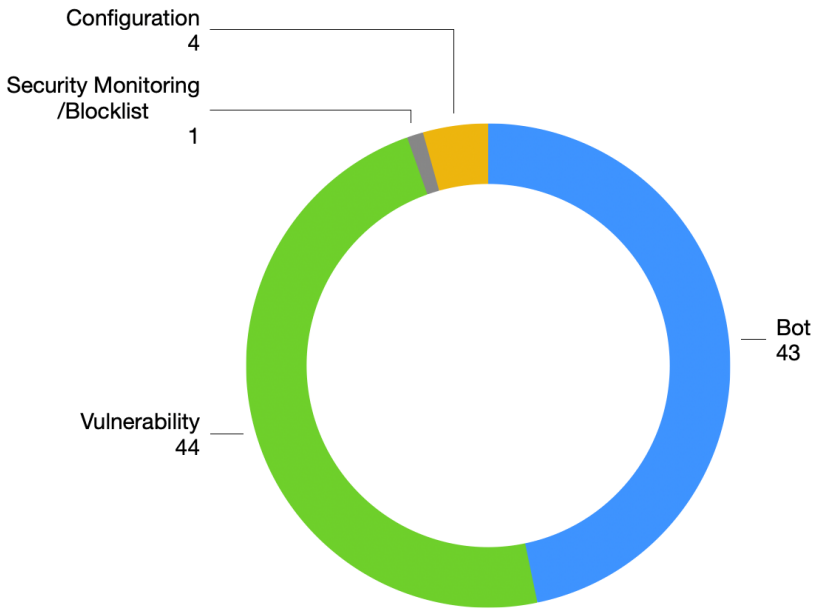

Abbildung 3.8: Anteil der Meldungstypen an den weiter zu bearbeitenden Warnmeldungen

An dieser Stelle könnte auch noch einmal darauf eingegangen werden, wie viele Warnmeldungen als Wiederholungsmeldungen identifiziert werden, aber als ursprüngliche Warnmeldung eine Meldung haben, die älter als 90 Tage ist. Hier wurden tatsächlich kaum Einträge gefunden, die diesen Zeitraum überschreiten. Das bedeutet allerdings nicht notwendigerweise, dass die Schwachstellen oder Ähnliches zweifelsfrei behoben sind. Es heißt nur, dass unter den betrachteten Daten keine neueren Warnmeldungen darauf hindeuten.

So kann es beispielsweise sein, dass das DFN-CERT ebenfalls nur für einen bestimmten Zeitraum Warnmeldungen für eine Schwachstelle verschickt. Die Schwachstelle könnte also nach wie vor existieren - es gehen lediglich keine neuen AW-Meldungen mehr ein. So sind beispielsweise 129 Warnmeldungen zu einer Vulnerability/Jenkins-Schwachstelle zwischen dem 10.02.2024 und dem 05.07.2024 eingegangen. Da die Meldungen zuvor beinahe täglich eingingen und danach keine mehr auftaucht, könnte dies auf eine behobene Schwachstelle schließen. Zugleich fällt das Enddatum aber auch genau in die Woche der Inventarscans von Schwachstellen des DFN-CERTs. Es könnte somit auch mit diesen Inventarscans einen Wechsel in den zu meldenden Schwachstellen gegeben haben, was den anschließenden Stop der Warnmeldungen erklären könnte.

Des Weiteren gibt es mit dem Beispiel des Blocklist-Eintrags eine Wiederholungsmeldung, die sich bis ans Ende der Datensätze mit dem 10.09.2024 vermutlich über die vorhandenen Datensätze hinauszieht. Es braucht somit weitere Daten, um herauszufinden, ob diese Warnmeldung nach wie vor gesendet wird.

Wie mit den nun resultierenden Daten fortgefahren werden kann, wird im folgenden Kapitel weiter beleuchtet.

4 Playbooks

Nachdem nun festgestellt wurde, welche Daten tatsächlich vorliegen, gilt es zu betrachten, wie mit den übrigen Warnmeldungen umgegangen werden sollte. Wie in Abbildung 3.8 beschrieben wurde, gibt es nach den Filterprozessen noch Meldungen der Kategorien *Bot*, *Vulnerability*, *Security Monitoring* und *Configuration*. Da dies somit die relevantesten Meldungstypen sind, wäre es sinnvoll, für jeden dieser Typen einen spezifischen Prozess zu erstellen und somit ein klares Playbook für diese Vorfälle zu haben.

Bisher gibt es lediglich einen generischen Incident-Handling-Prozess an der HAW Hamburg. Dieser wird in [12] genauer beschrieben. Dieser bestehende Prozess beinhaltet jedoch einige Nachteile. Da er beispielsweise so generisch gehalten ist, bietet er keinen genauen Überblick darüber, welche konkreten Aufgaben im Falle von spezifischen Incidents abgearbeitet werden müssen. Dies birgt das Risiko, dass Aufgabenteile bei der Bearbeitung eines Sicherheitsvorfalls leichter vergessen werden können. Zudem kostet es zusätzliche Zeit, die entsprechenden Aufgaben im Ernstfall herauszufinden. Diese Bearbeitungsdauer kostet nicht nur Ressourcen, sondern kann bei der Bearbeitung von sicherheitskritischen Vorfällen auch entscheidend sein.

Zudem ist der jetzige Prozess sehr starr. Bestimmte Incidents benötigen eventuell keine umfangreiche Response und könnten in kleinerem Rahmen gelöst werden. In der Art und Weise, wie der generische Incident-Handling-Prozess strukturiert ist, bietet sich hierfür nicht die nötige Flexibilität und für kleinere Incidents werden unnötige Ressourcen aufgewendet.

Ein weiterer Vorteil von klar definierten Prozessen ist es, dass sie eine Grundvoraussetzung dafür darstellen, den gesamten Prozess oder auch Teilprozesse zu automatisieren. Ist dies nicht der Fall, gibt es keinen klaren Überblick über die einzelnen Aufgaben, die für eine Automatisierung in Frage kommen.

Im folgenden Abschnitt werden zunächst grobe Konzepte für Playbooks vorgestellt, die dann im weiteren Verlauf des Kapitels als Grundlage für die tatsächlichen Prozesse dienen

können. Diese Prozesse werden dann genauer durch Prozessdiagramme beschrieben und anhand dieser näher erläutert.

4.1 Playbook-Konzepte

Die Basis für die folgenden Playbook-Konzepte bietet bereits [15], wo begonnen wurde, Playbook-Konzepte auszuarbeiten, die dann als Grundlage für die spezifischen Prozesse dienen können. Da nun feststeht, welche Meldungstypen einen spezifischen Prozess benötigen, wurde die Liste der Playbook-Konzepte um diese erweitert. Die ursprünglichen Konzepte (die Playbooks und Sub-Playbooks *Main*, *Bot*, *Bot/HTTP* und *Vulnerability*) wurden, nach einigen kleineren Änderungen, aus [15] übernommen - um an dieser Stelle eine vollständige Liste der Konzepte vorzustellen.

Es gilt zu beachten, dass im folgenden Abschnitt nicht die fertigen Prozesse vorgestellt werden. Es handelt sich bei den Konzepten auch nicht um eine Kurzbeschreibung der resultierenden Prozesse. Dieses gesamte Kapitel soll einen Ansatz für den Prozess darstellen, der durchlaufen werden kann, um Playbooks zu erstellen. Ein Teilschritt hierbei kann es sein, sich in erster Instanz Konzepte zu überlegen, die dann im Anschluss konkreter ausgearbeitet und zu tatsächlichen Prozessdiagrammen und Playbooks transformiert werden können. Es handelt sich bei den Playbook-Konzepten also nur um ein unfertiges Produkt, was angeführt wird, um die Erstellung von Playbooks zu veranschaulichen. Das fertige Endergebnis wird dann mit Abschnitt 4.2 vorgestellt und weicht bisweilen deutlich von den Playbook-Konzepten ab.

4.1.1 Playbook-Nummer 1.0.0 - Main-Playbook für AW-Messages

Dieses Playbook stellt den Hauptprozess dar, der automatisiert gestartet wird, wenn eine Warnmeldung eingeht. Dabei stehen Datenverarbeitung, Filterung und Priorisierung im Vordergrund. Weitere generische Aufgaben wie die Ticketerstellung werden ebenfalls abgearbeitet.

- **Automatisierte Datenverarbeitung und Extraktion:** Warnmeldung wird automatisiert eingelesen; Werte von relevanten Feldern (z.B. Datum, Kategorie oder betroffene Ressource) werden extrahiert; Verifizierung der Datenintegrität; Daten werden für die Weiterverarbeitung formatiert

- **Filterprozess - Relevanzprüfung:** Anwendung der ersten Welle von Filtern, die bestimmen, ob die Warnmeldung irrelevant ist (z. B. doppelte Meldungen, bekannte Dienste); Archivierung aller irrelevanten Meldungen und Entfernen aus dem Incident-Response-Prozess
- **Filterprozess – Prioritätsbewertung:** Hier werden Filter ausgeführt, die Warnmeldungen in hohe oder niedrige Priorität unterteilen. Konkret wird im Bedarfsfall ein Dringlichkeitsvermerk der Warnmeldung hinzugefügt, der die Priorisierung ausdrückt.
- **Kategorisierung nach Meldungstyp:** Automatische Zuordnung der Warnmeldung zu einem Meldungstyp (z. B. *Bot* oder *Vulnerability*); Überprüfung, ob die Warnmeldung einer spezifischen Unterkategorie zugeordnet werden kann (z. B. *Bot/HTTP*)
- **Start des spezifischen Playbooks:** Basierend auf dem ermittelten Meldungstypen wird das zugehörige Playbook gestartet.
  - *Bot:* Sub-Playbook 1.1.0 wird gestartet.
  - *Vulnerability:* Sub-Playbook 1.2.0 wird gestartet.
  - *Security Monitoring/Blocklist:* Sub-Playbook 1.3.0 wird gestartet.
  - *Configuration:* Sub-Playbook 1.4.0 wird gestartet.
- **Ticket-Erstellung:** Erstellung eines Incident-Tickets für die Warnmeldung zur Verfolgung der Bearbeitung des Vorfalls; Ticket wird automatisch an das Incident-Response-Team weitergeleitet
- **Berichterstattung und Archivierung:** Zusammenfassung der Ergebnisse und Dokumentationen der Bearbeitung (z.B. getroffene Maßnahmen) in einem Bericht; Archivierung der AW-Meldung

4.1.2 Playbook-Nummer 1.1.0 - Sub-Playbook: Bot

Das folgende Playbook wird genutzt, falls eine AW-Meldung eingeht, die besagt, dass ein System als Teil eines Botnets identifiziert wurde.

- **Netzwerkisolierung:** Kompromittiertes System im Netzwerk wird umgehend isoliert, um weitere Verbindungen zum Botnet-Control-Server zu unterbinden.
- **Untersuchung des kompromittierten Systems:** Analyse des Systems: gibt es Hinweise auf Bot-Software oder weitere Kompromittierung?
- **Ticketanreicherung durch CERT DIAGNOSIS:** Falls im Feld *CERT DIAGNOSIS* der Name eines Botnets angegeben ist, wird das Ticket automatisch um zusätzliche Informationen zum Botnet angereichert (beispielsweise durch Abrufen von Daten von externen Quellen oder Datenbanken).
- **Start eines spezifischen Sub-Sub-Playbooks:** Falls die Warnmeldung einer Unterkategorie wie *Bot/HTTP* zugehört und existiert dafür ein spezifisches Sub-Sub-Playbook, wird dieses hier gestartet. Auch wenn im vorherigen Schritt spezifische Informationen zu dem speziellen Typ des Botnets aus dem Feld CERT DIAGNOSIS gezogen werden konnten, kann es hier in Zukunft weitere Sub-Playbooks geben, die dafür an dieser Stelle gestartet werden können.
  - *Bot/HTTP:* Sub-Sub-Playbook 1.1.1 wird gestartet.
- **Beseitigung des Bots (wenn kein Sub-Sub-Playbook gestartet wurde):** Basierend auf den vorliegenden Informationen werden weitere Maßnahmen eingeleitet, die die Bot-Software vom System entfernen; Tests, die die vollständige Bereinigung des Systems prüfen.
- **Präventive Maßnahmen:** Implementierung von präventiven Maßnahmen wie der Erstellung eines eigenen Sub-Playbooks zur Bearbeitung künftiger, gleicher Bots

Playbook-Nummer 1.1.1 - Sub-Sub-Playbook: Bot/HTTP

Dieses Playbook dient der Bearbeitung von Warnmeldungen, bei denen festgestellt wurde, dass die Kommunikation mit dem Botnet-Control-Server über eine HTTP-Verbindung erfolgte.

- **Analyse der HTTP-Kommunikation:** HTTP-Anfragen werden zur Identifizierung verdächtiger Requests untersucht, die auf eine Kommunikation mit einem Botnet-Control-Server hinweisen können; HTTP-Header und Request-Bodys werden geprüft, um festzustellen, welche Daten gesendet oder empfangen wurden.

- **Logdatenanalyse:** Auch die Systemlogs werden auf Hinweise zu versendeten oder empfangenen Daten geprüft
- **Zielserver-Validierung:** Zielserver wird analysiert; bei verdächtigem Zielserver wird die HTTP-Kommunikation blockiert
- **Netzwerkisolierung und Systembereinigung:** Zur Sicherstellung, dass keine weitere Kommunikation zwischen dem System und dem Botnet-Control-Server möglich ist, wird die Netzwerkisolierung bis zur Systembereinigung aufrechterhalten.
- **Detaillierte Berichterstattung:** Detaillierter Bericht über getroffene Maßnahmen und Kommunikation über HTTP, Forensische Schritte werden dokumentiert

4.1.3 Playbook-Nummer 1.2.0 - Sub-Playbook: Vulnerability

Dieses Playbook wird für die Bearbeitung von Warnmeldungen zu Schwachstellen auf Systemen eingesetzt.

- **Erkennung und Validierung der Schwachstelle:** Schwachstelle wird geprüft (z. B. durch CVE-Nummern oder bekannte Schwachstellen); Validierung, ob die Schwachstelle tatsächlich auf dem System besteht (beispielsweise durch Überprüfen der Softwareversion oder bestehenden Konfigurationen)
- **Beseitigung der Schwachstelle:** Es werden Schritte für die Beseitigung der Schwachstelle eingeleitet. Gibt es beispielsweise verfügbare Patches oder Updates für die betroffene Schwachstelle, werden diese für die Software eingespielt.
- **Forensische Analyse:** Untersuchung des Systems auf mögliche Hinweise, ob die Schwachstelle bereits ausgenutzt wurde (beispielsweise in Logs oder weiteren verdächtigen Aktivitäten); gibt es Hinweise auf eine Kompromittierung des Systems, können weitere Playbooks (beispielsweise für eine detaillierte Untersuchung) gestartet werden und das Incident-Response-Team wird alarmiert
- **Präventive Maßnahmen:** Maßnahmen zur Vermeidung ähnlicher Schwachstellen werden für die Zukunft implementiert (zum Beispiel regelmäßige Sicherheitsupdates und Überprüfungen der Konfigurationen); Einführung von spezifischen Penetrationstests für ähnliche Sicherheitslücken

4.1.4 Playbook-Nummer 1.3.0 - Sub-Playbook: Security Monitoring/Blocklist

Geht eine Warnmeldung vom Typ *Security Monitoring/Blocklist* ein, wird das folgende Playbook eingesetzt. Es handelt sich dabei um ein Sub-Playbook und kein Sub-Sub-Playbook, da keine weiteren Warnmeldungen des Typs *Security Monitoring* eingegangen sind. Trotz der spezifischeren Unterkategorie wurde somit ein Sub-Playbook gewählt.

- **Prüfung von öffentlichen Listen:** Es wird geprüft, auf welcher öffentlichen Blockliste die betroffene Ressource aufgeführt ist.
- **Untersuchung der Ursachen:** Wieso wurde die betroffene Ressource auf eine Blockliste gesetzt?
- **Eventuell Kontaktaufnahme mit Betreiber*innen:** Je nach Ursache ist es möglich, die Betreiber*innen der Blockliste zu kontaktieren, um sich von der Liste entfernen zu lassen. Gibt es keine Reaktion, bleibt auch die Möglichkeit, Kontakt zum DFN-CERT aufzunehmen, um die betroffene Ressource lokal davon auszunehmen, sodass keine weiteren Warnmeldungen folgen.

4.1.5 Playbook-Nummer 1.4.0 - Sub-Playbook: Configuration

Dieses Playbook dient der Bearbeitung von Warnmeldungen, bei denen unerwünschte Konfigurationen von Systemen gemeldet werden. Diese Konfigurationen könnten den Zugang zu Teilen des Systems erleichtern oder Angriffe ermöglichen.

- **Erkennung und Validierung der unerwünschten Konfiguration:** Verifizierung der gemeldeten Konfiguration und Überprüfung, ob diese potenziell ausnutzbar ist
- **Start eines spezifischen Sub-Sub-Playbooks:** Falls die Warnmeldung einer Unterkategorie wie *Configuration/Unrestricted Access* angehört und existiert dafür ein spezifisches Sub-Sub-Playbook, wird dieses hier gestartet.
  - *Configuration/Unrestricted Access:* Sub-Sub-Playbook 1.4.1 wird gestartet.
  - *Configuration/Unsafe Cryptography:* Sub-Sub-Playbook 1.4.2 wird gestartet.

- **Konfigurationsänderungen (wenn kein Sub-Sub-Playbook gestartet wurde):** Die fehlerhafte Konfiguration wird in diesem Schritt überprüft und entsprechende Anpassungen werden an der Konfiguration vorgenommen, um die Sicherheitslücke zu schließen.
- **Forensische Analyse:** Untersuchung des Systems auf mögliche Ausnutzung der fehlerhaften Konfiguration (z. B. Protokollanalyse oder Überprüfung von Logs)
- **Berichterstattung:** Dokumentation der durchgeführten Änderungen an der Konfiguration und welche potenzielle Bedrohung von dieser ausging
- **Präventive Maßnahmen:** Regelmäßige Überprüfung der Konfigurationen und Implementierung von Sicherheitsrichtlinien, um ähnliche Konfigurationsfehler in der Zukunft zu verhindern

Playbook-Nummer 1.4.1 - Sub-Sub-Playbook: Configuration/Unrestricted Access Dieses Playbook dient der Bearbeitung von Vorfällen, bei denen ein Dienst auf einem System ohne ausreichende Zugriffskontrolle betrieben wird.

- **Erkennung und Validierung der Zugriffsbeschränkung:** Verifizieren, ob der gemeldete Dienst tatsächlich ohne wirksame Authentifizierung zugänglich ist
- **Konfigurationsänderungen:** Implementierung von Authentifizierungsmechanismen und Zugriffsbeschränkungen, um den Dienst zu sichern (z. B. durch Firewalls oder Passwortschutz)

Playbook-Nummer 1.4.2 - Sub-Sub-Playbook: Configuration/Unsafe Cryptography Sollten Teile eines Systems eine unsichere Verschlüsselung nutzen, kann das folgende Playbook genutzt werden, um dieses Sicherheitsrisiko zu entfernen.

- **Erkennung und Validierung der kryptografischen Schwachstelle:** Überprüfung, ob das System tatsächlich unsichere kryptografische Methoden verwendet (z. B. schwache oder veraltete Verschlüsselungsalgorithmen)
- **Aktualisierung der Kryptografie:** Die genutzten kryptografischen Methoden auf dem System, die eine Schwachstelle darstellen, werden aktualisiert (beispielsweise durch Einspielen von Patches oder dem Austausch durch modernere Algorithmen).

4.2 Playbook-Prozesse

Der folgende Abschnitt versucht, die grob beschriebenen Playbook-Konzepte in klar strukturierte Prozessdiagramme zu überführen. Dabei gab es an diversen Stellen Änderungen, anderes wurde weggelassen und neue Aufgaben kamen hinzu.

Der Ansatz hierbei war es jedoch nicht, für sich alleinstehende Playbooks zu entwickeln, die dann genutzt werden, wenn ein entsprechender Incident eintritt. Stattdessen war die Überlegung, dass es sinnvoller wäre, den bereits bestehenden, generischen Incident-Handling-Prozess der HAW Hamburg zu erweitern. Somit könnten sich die spezifischen Reaktionen auf konkrete Warnmeldungen direkt in den allgemeineren Prozess einreihen und es wäre klar nachvollziehbar, wann welcher Unterprozess genutzt werden sollte.

4.2.1 Generischer Incident-Handling-Prozess

Ein erster Schritt ist es, ein Verständnis für die Arbeit in [12] zu schaffen. Diese stellt den generischen Incident-Handling-Prozess der HAW Hamburg vor und somit den Prozess, der durch diese Arbeit um spezifische Unterprozesse erweitert werden soll.

Phasen

Bereits 2015 wurden erste Bestrebungen unternommen, einen Incident-Handling-Prozess für die HAW Hamburg zu entwerfen. [12] nimmt diesen als Grundlage für die Arbeit und entwirft darauf basierend einen verbesserten Ansatz. Der neue Ansatz lässt sich in vier Phasen einteilen, welche in Abbildung 4.1 visualisiert sind und im folgenden Abschnitt kurz vorgestellt werden. Für nähere Erläuterungen gilt hier der Verweis auf [12].

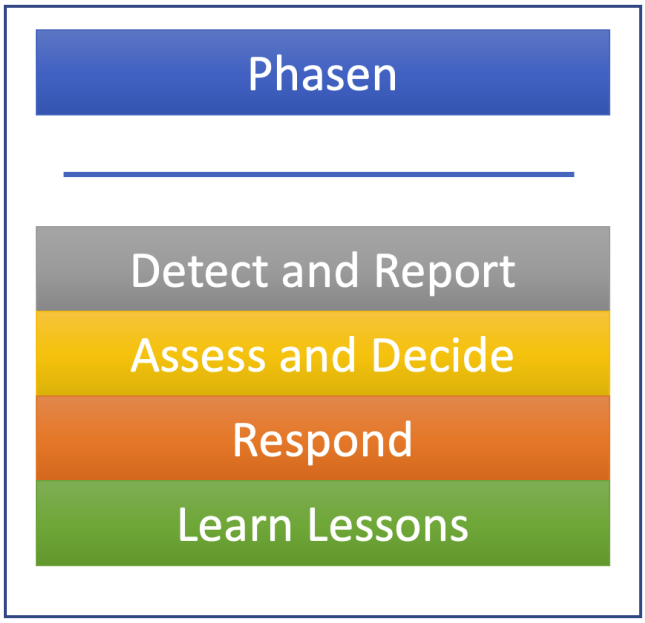

Abbildung 4.1: Phasen des generischen Incident-Handling-Prozesses der HAW Hamburg - eigene Darstellung nach [12]

- **Detect and Report:** Diese Phase dient der Identifizierung von Sicherheitsvorfällen. Verschiedene Quellen können als Ausgangspunkte der Identifizierung gelten und es wird entschieden, ob es sich um einen Fehlalarm handelt, oder tatsächlich ein potenzieller Incident vorliegt. Darauf basierend wird dann entweder das Incident Handling beendet oder die Phase *Asses and Decide* angestoßen.
- **Assess and Decide:** Hier folgt nun zunächst eine Triage, um den Schweregrad des Incidents zu bestimmen, woraus sich die notwendigen Maßnahmen ableiten lassen. Liegt ein Incident vor, müssen Meldepflichten erfüllt werden; es werden Zuständigkeiten geprüft und eine geeignete Response wird formuliert. Sollte es sich um eine Crisis handeln, wird in diesem Schritt der Incident dem Krisenmanagement übergeben.
- **Respond:** Im Anschluss wird in der Phase *Respond* die tatsächliche Response ausgeführt. Hier werden Aufgaben zwischen verschiedenen Beteiligten aufgeteilt und delegiert.

- **Learn Lessons:** Nach Abschluss der Response dient die Phase *Learn Lessons* dazu, Maßnahmen zu identifizieren, um ähnliche Vorfälle in Zukunft zu verhindern und die bestehenden Prozesse dahingehend zu optimieren.

Rollen

Der generische Incident-Handling-Prozess besitzt eine Reihe von Rollen mit unterschiedlichen Verantwortlichkeiten. Für diese Arbeit besonders relevant sind die Rollen *Incident Coordinator*, *Incident Responder* und *CISO*.

Incident Coordinator Bei dem Incident Coordinator handelt sich um die zentrale Führungsperson von einem oder mehreren Incident-Response-Teams. Zu den Hauptaufgaben zählen die Evaluierung von potenziellen Sicherheitsvorfällen und die Koordination des Teams. Der Incident Coordinator dient zudem als zentrale Kommunikationsschnittstelle mit internen und externen Organisationen sowie dem ITSC (Informationstechnik Service Center) [13].

Incident Responder Ein Incident Responder ist die zentrale Figur für die Bearbeitung der Sicherheitsvorfälle. Sie werden in Incident-Response-Teams zusammengefasst. Die tatsächliche Zusammensetzung und Größe dieser Teams kann je nach Vorfall stark variieren [13]. Da diese Teams lediglich temporär sind und meist nur für die Dauer der Bearbeitung eines Incidents bestehen, werden sie im Folgenden auch als Ad-hoc Teams bezeichnet [12].

CISO Für das Incident Management der HAW Hamburg stellt der CISO (Chief Information Security Officer) eine zentrale Position dar. Besonders relevant ist dieser für die Koordinierung und Analyse von Sicherheitsvorfällen. Neben der Untersuchung von Incidents gilt auch die Erarbeitung von Sicherheitskonzepten zu den zentralen Aufgaben eines CISOs. Während ein Incident Coordinator die zentrale Rolle für die Koordination eines bestimmten Vorfalls übernimmt, ist der CISO die zentrale Position über alle Sicherheitsvorfälle hinweg [12].

Die weiteren Rollen können in den beiden Projekten [13] und [12] nachgelesen werden. Für einen kurzen Überblick kann die Visualisierung in Abbildung 4.2 aus [12] genügen.

Die meisten Rollen sind jedoch für die vorliegende Arbeit weniger relevant und für eine nähere Erläuterung der Grafik wird somit auf die zugrundeliegende Arbeit verwiesen.

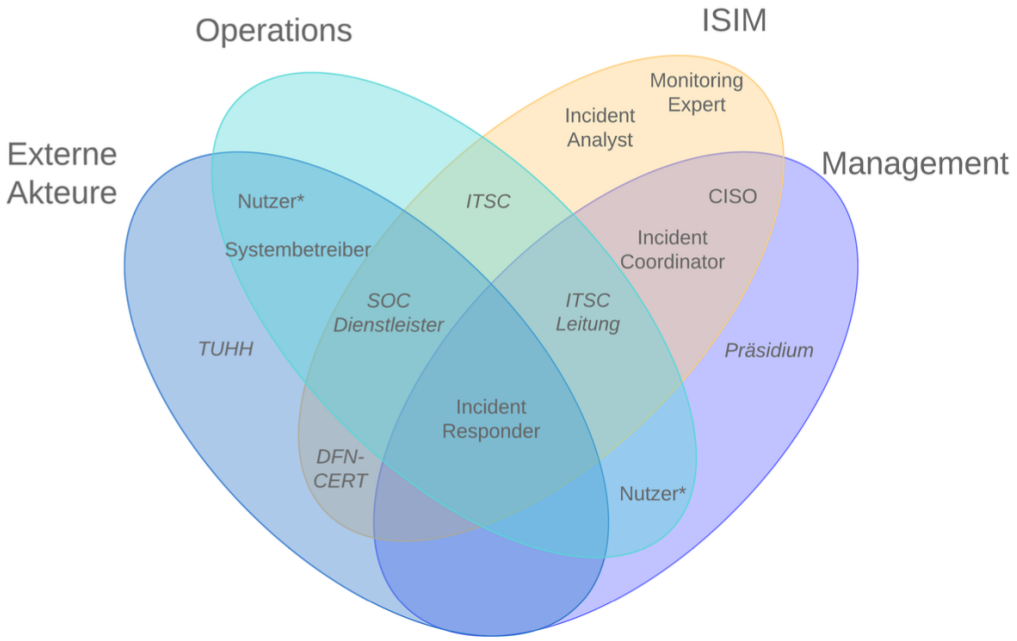

Abbildung 4.2: Übersicht der Rollen im generischen Incident-Handling-Prozess der HAW Hamburg - aus [12]

4.2.2 Spezifische Playbooks

Die spezifischen Prozesse erweitern nun diesen bereits bestehenden, generischen Prozess. Hierzu wurden die spezifischen Prozesse in BPMN (Business Process Model and Notation) modelliert. Die Grundidee hierbei ist, dass je nach Warnmeldungstyp ein eigenständiger Unterprozess angestoßen werden kann. Dies bietet die oben beschriebenen Vorteile der konkreteren Beschreibung von Aufgaben und der Möglichkeit für eine schlankere Bearbeitung.

Vorweg sollte jedoch auch ein Blick auf mögliche Nachteile des neuen Ansatzes geworfen werden. Speziell sticht hier hervor, dass die bestehende, klare Einteilung des Prozesses in die vier Phasen aufgebrochen wird. Die neuen Unterprozesse gehen nun über mehrere

Phasen hinweg. Insbesondere die Phasen *Asses and Decide* und *Respond* werden nun zusammengefasst. Der Grund hierfür ist, dass bereits während der ersten Phase *Detect and Report* eine Vorverarbeitung stattfindet und der Evaluationsprozess dadurch erheblich reduziert wird. Somit wurde sich bei der vorliegenden Arbeit auf ein Prozessdiagramm pro Meldungstyp beschränkt. Für zukünftige Arbeiten kann es jedoch durchaus sinnvoll sein, auch für die spezifischen Playbooks eine Trennung nach den bestehenden Phasen durchzuführen. Dies wäre insbesondere ratsam, wenn die Unterprozesse weiter verfeinert werden und somit in ihrer Komplexität wachsen.

Ein weiterer Nachteil des gewählten Ansatzes ist die Menge an Dopplungen zwischen den verschiedenen, spezifischen Prozessen und auch dem generischen Prozess. Dies begründet sich unter anderem damit, dass beispielsweise auch von einem spezifischen Prozess ein generischer Prozess gestartet können werden muss, wenn weitere sicherheitsrelevante Informationen während der Bearbeitung des Incidents ans Licht kommen. Auch weitere Aufgaben wie die Notwendigkeit von Meldepflichten müssen bei mehreren Meldungstypen beachtet werden. Eine Extraktion solcher Unterprozesse könnte ebenfalls in Zukunft von Nutzen sein, um bei steigender Komplexität der Prozesse die Übersichtlichkeit zu gewährleisten. Für die vorliegende Arbeit wurde sich darauf beschränkt, bei der Visualisierung die Teile auszugrauen, die bereits in anderen Prozessen abgedeckt sind und lediglich die Teile des Prozessdiagramms hervorzuheben, die Neuerungen darstellen. Dies hat den Vorteil, dass gleiche Inhalte schnell erfasst werden können, die spezifischen Prozesse allerdings trotzdem als klare Folge von Aufgaben in einem Prozessdiagramm dargestellt werden.

Detect and Report

Abbildung 4.3 zeigt die Erweiterung der Phase *Detect and Report* um eine zusätzliche Swimlane 'Vorbearbeitung'. Wie der Name schon andeutet, werden hier die automatischen Warnmeldungen, die von extern an die HAW Hamburg gesendet werden, vorverarbeitet.

Hauptziel dieser Änderung ist es, die Filtermöglichkeiten, die in [15] und in Abschnitt 3 vorgestellt wurden, in den bestehenden Prozess zu integrieren. Hierzu müssen zunächst die Daten aus den eingehenden Warnmeldungen extrahiert und verarbeitet werden. Anschließend folgt die erste Filterwelle, die prüft, welche Relevanz die vorliegende Warnmeldung hat. Handelt es sich dabei um eine Fehlmeldung oder ist sie generell irrelevant,

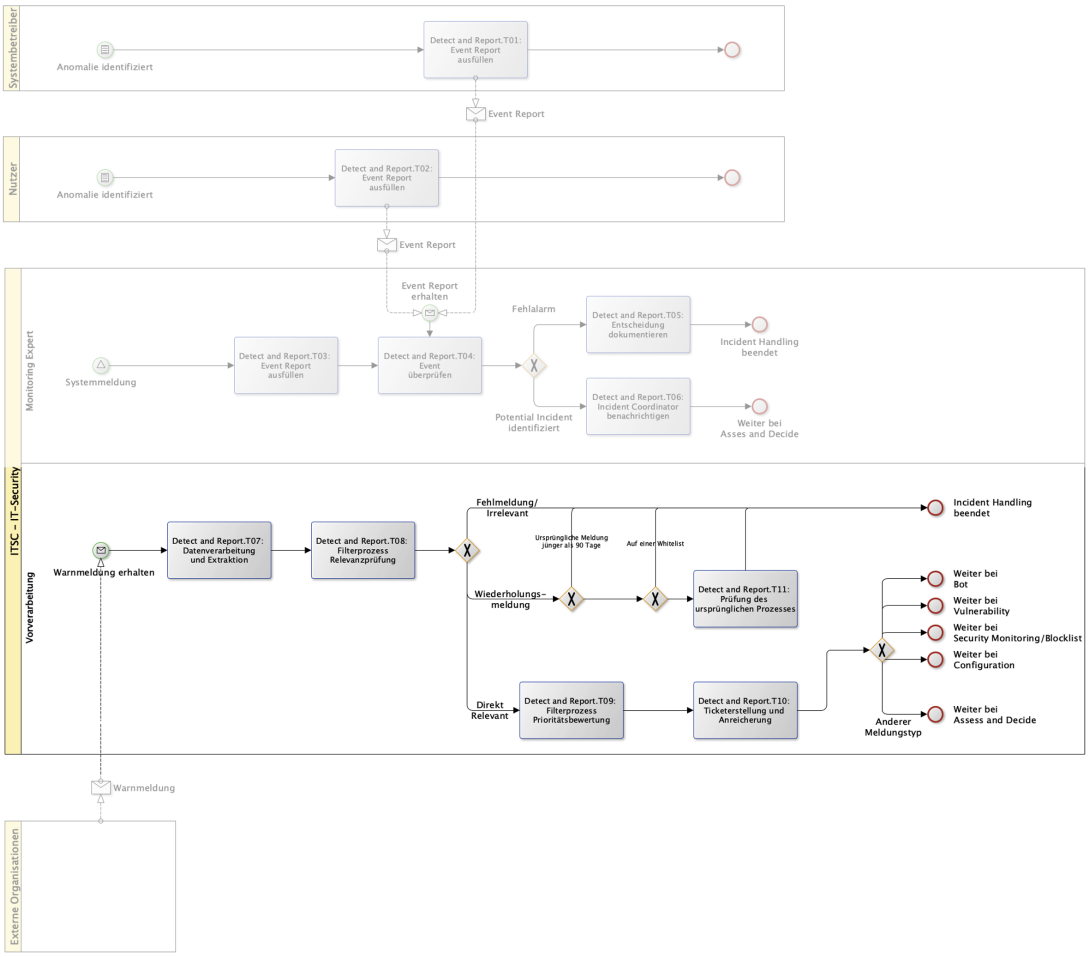

Abbildung 4.3: Prozess: Detect and Report - verändertes Diagramm aus [12]

kann das Incident Handling direkt beendet werden. Sind sie unmittelbar relevant, folgt die zweite Reihe an Filtern, um die Priorität zu bestimmen, die schließlich im Ticket, das im Anschluss erstellt wird, vermerkt werden kann. Das Ticket kann an dieser Stelle zudem auch mit weiteren Informationen angereichert werden. Eine Möglichkeit hierfür wäre etwa die Angabe einer CERT DIAGNOSIS. Dies könnte sogar erweitert werden, wenn beispielsweise hier der Name eines Botnetzes aufgeführt wird. Dabei könnte es sinnvoll sein, das Ticket um Informationen zu diesem Botnetz zu erweitern. Warnmeldungen mit dem Meldungstyp *Vulnerability* bieten häufig innerhalb der zusätzlichen Informationen die CVE-Nummer an, mit deren Hilfe ebenfalls aus weiteren Quellen nützliche Hinweise für die Bearbeitung des Tickets gewonnen werden können.

Im Anschluss wird geprüft, ob ein spezifischer Prozess für den besagten Meldungstypen existiert. Ist dies der Fall, wird dieser Prozess gestartet. Andernfalls wird an dieser Stelle an den generischen Prozess verwiesen.

Ein besonderes Augenmerk sollte zudem auf eine weitere Option nach dem ersten Filterprozess geworfen werden. Die Möglichkeit der 'Wiederholungsmeldung'. Wie in Abschnitt 3.2.2 beschrieben wird, gibt es eine Vielzahl an Meldungen, die lediglich wiederholte Warnungen darstellen. Wenn beispielsweise eine bestimmte Schwachstelle weiter existiert, wird die gleiche Warnmeldung erneut versendet, um darauf hinzuweisen. Ist hierfür bereits ein Incident-Response-Prozess gestartet, ist es nicht sinnvoll, einen weiteren für die gleiche Aufgabe zu starten. Stattdessen wird zunächst geprüft, ob die ursprüngliche Meldung jünger als 90 Tage ist. Auch wenn die Ursache auf einer Whitelist vermerkt ist, wird das Incident Handling beendet. Eine solche Whitelist wird beispielsweise für bestimmte Konfigurationen geführt, bei denen bereits eine Abwägung stattgefunden hat und festgestellt wurde, dass die Risiken den Nutzen nicht übersteigen. Diese Whitelists müssen regelmäßig geprüft werden, um die Aktualität zu gewährleisten. Ist die ursprüngliche Meldung älter als 90 Tage und nicht auf einer Whitelist, wird der ursprüngliche Prozess noch einmal aktiv angestoßen und untersucht, aus welchem Grund die Bearbeitung noch nicht zur Gänze erfolgt ist. So kann sichergestellt werden, dass es weniger Fälle von unbearbeiteten Prozessen gibt und Wiederholungsmeldungen nicht einfach durchrutschen.

An einigen Stellen zeigen sich hier Abweichungen vom ursprünglichen Playbook-Konzept aus 4.1.1. Besonders auffallend ist hier die Einführung des Unterprozesses deer Wiederholungsmeldungen. Dieser ist im Main-Playbook noch gar nicht aufgeführt. Die Berichterstattung und Archivierung wurde hingehen an andere Stellen verlagert, um die bisherige Notation mit dem Übergang in andere Prozessdiagramme beizubehalten. Die Ticketerstellung vor der Ausführung der spezifischen Playbooks zu veranlassen, ergibt letzlich deutlich mehr Sinn, um den Vorverarbeitungsprozess vor dem Start der Sub-Playbooks abzuschließen.

Bot

In Abbildung 4.4 findet sich das Prozessdiagramm zum Sub-Playbook Bot. Konkret enthält dieses sogar auch das Sub-Sub-Playbook Bot/HTTP. Im Rahmen der Erstellung des Prozessdiagramms hat sich herauskristallisiert, dass der einzige richtige Unterschied

zwischen den beiden Playbook-Konzepten die spezifische Fokussierung auf eine HTTP-Kommunikation war. Da aber der generischere Meldungstyp Bot eine Kommunikation über HTTP nicht ausschließt, sollte die HTTP-Kommunikation sowohl für Bot als auch für die Unterkategorie Bot/HTTP untersucht werden. Ähnlich verhält es sich mit den restlichen Analysen wie der Logdatenanalyse und der Analyse des Zielservers. Beides sollte für beiden Meldungskategorien vorgenommen werden. Insofern ergibt eine Trennung der Prozessdiagramme keinen Sinn und selbst innerhalb desselben Prozesses sollten die gleichen Aufgaben bearbeitet werden. Bei nachfolgenden Arbeiten könnte die erneute Trennung sinnvoll sein, sollten sich unterschiedliche Aufgabenbereiche für die unterschiedlichen Kategorien ergeben.

Nach Eingang des Tickets zur Warnmeldung startet der Prozess mit einer Netzwerkisolierung. Dieses ist zwingend erforderlich, um das kompromittierte System zu isolieren und weitere Verbindungsversuche zum Botnet-Control-Server zu unterbinden. Im Anschluss darauf folgt eine erste Untersuchung des Systems. Die Warnmeldung selbst dient nur als Warnhinweis, während nun tatsächlich auf dem System verfolgt wird, ob es Hinweise auf eine Bot-Software gibt. Bei einem Fehlalarm wird an die Phase *Learn Lessons* übergeben. Gibt es Anzeichen auf eine Kompromittierung des Systems, wird der Schweregrad geprüft. Bei einer Crisis folgt die Benachrichtigung des CISOs und es wird an das Krisenmanagement übergeben. Sonst wird mit Rücksprache mit dem CISO ein Ad-hoc Team bestimmt, das die weitere Bearbeitung des Vorfalls übernimmt. Nach einigen Analyseschritten wird erneut mit dem Incident Coordinator Rücksprache gehalten, ob eine Krise vorliegt. Ist dies nicht der Fall, wird vom Ad-hoc Team das System bereinigt. Im Anschluss werden Präventivmaßnahmen eingeleitet, um ähnliche Vorfälle in Zukunft zu verhindern und besser auf diese reagieren zu können. Dies könnte auch als Teil von Learn Lessons betrachtet werden, hier wurde sich jedoch aktiv dagegen entschieden, um den Fokus auf meldungstyp-spezifischere Maßnahmen zu lenken. Hierzu zählen beispielsweise das Aufsetzen von eigenen Sub-Playbooks für den konkreten Bot oder Bot-Typ, die womöglich eigene Schritte beinhalten können, um eine Bearbeitung in Zukunft zu vereinfachen. Auch bestimmte Sicherheitskonfigurationen und Testmöglichkeiten sollten ausgelotet werden, um zusätzliche Präventivmaßnahmen einzuleiten.

Eine klare Dopplung gibt es hier bei den Unteraufgaben zur Erfüllung der Meldepflichten. Diese decken sich direkt mit den entsprechenden Aufgaben aus *Assess and Decide* des generischen Prozesses.

Eine weitere Änderung zu der Phase *Respond* im Prozessdiagramm des generischen Prozesses ist der Verzicht auf den *Kollaborationspool*. In [12] ergänzt dieser den Incident Responder, um darauf aufmerksam zu machen, dass bestimmte Aufgaben beispielsweise nicht nur direkt innerhalb des Ad-hoc Teams sondern auch von externen Personen übernommen werden können. Dies ist in diesem Fall für die Bearbeitung der Aufgabe nicht wahrscheinlich. Insofern wurde sich für die Rolle des Incident Responder entschieden. Sollten in Zukunft für konkrete Aufgaben weitere Personen außerhalb des Ad-hoc Teams zuständig sein, kann eine solche, weitere Trennung wieder eingeführt werden.

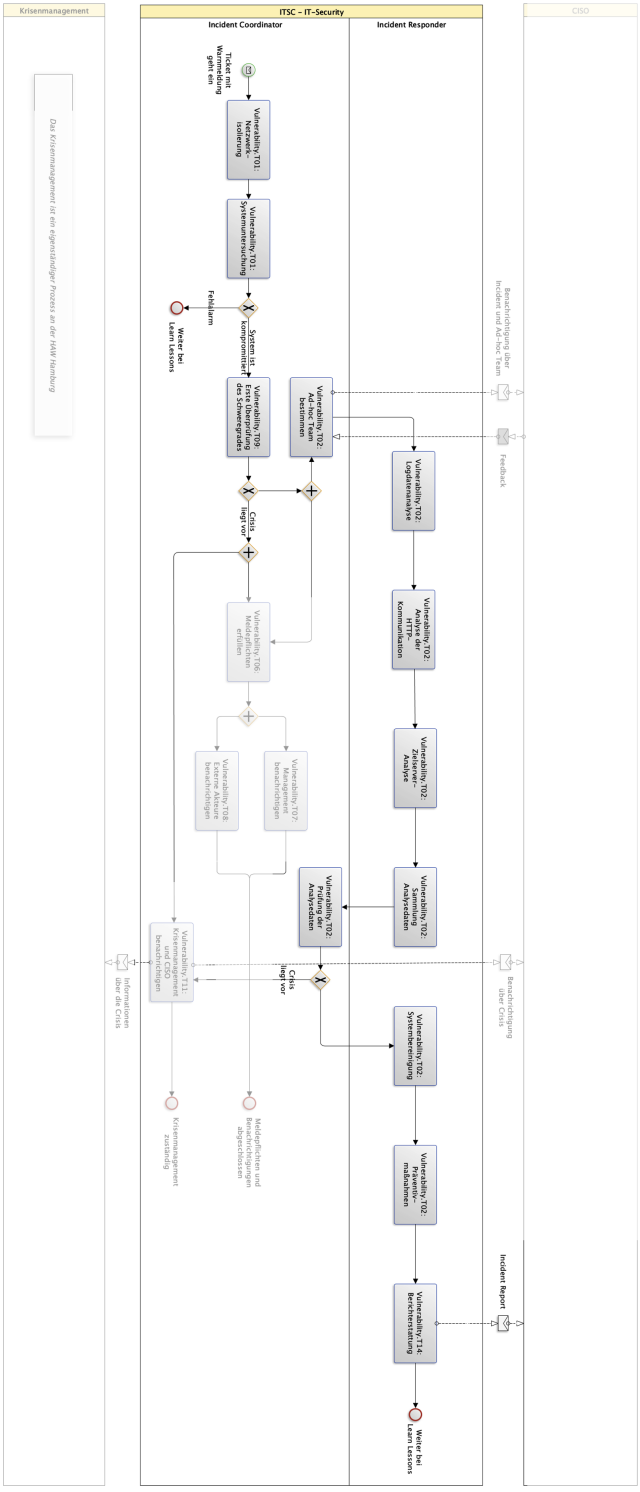

Abbildung 4.4: Prozess: Bot

Vulnerability

Der in Abbildung 4.5 gezeigte, spezifische Prozess für den Meldungstyp Vulnerability scheint sich auf den ersten Blick in vielen Teilaufgaben mit denen des generischen Incident-Handling-Prozesses zu decken. Der Grund hierfür ist relativ simpel. Bei einer Schwachstelle ist die kritische Frage, ob diese Schwachstelle bereits ausgenutzt wurde oder nicht. Ist dies nicht der Fall, müssen lediglich Maßnahmen eingeleitet werden, um die Schwachstelle zu beseitigen. Wurde sie hingegen ausgenutzt, muss der generische Prozess des Incident Handlings angestoßen werden, um den daraus resultierenden Sicherheitsvorfall zu bearbeiten. Eventuell ergibt sich hieraus auch direkt eine Crisis. In jedem Fall ist der Vulnerability-Prozess allein hierfür nicht mehr ausreichend.

Etwas irritierend kann die Tatsache wirken, dass die Aufgabe für die präventiven Maßnahmen parallel zur forensischen Analyse gestartet wird. Man könnte meinen, dass diese Aufgabe erst kurz vor Abschluss des Prozesses bearbeitet werden sollte. Fest steht jedoch, dass, egal was die forensische Analyse ergibt, es in jedem Fall umgehend notwendig ist, präventive Maßnahmen einzuleiten, um ähnliche Schwachstellen zu verhindern. Auch eine parallele Bearbeitung während dieser Analyse kann hierfür sinnvoll sein, weswegen dies im Prozessdiagramm auf diese Weise veranschaulicht wurde. Eine ausführliche Berichterstattung um die ergänzten Maßnahmen kann hierbei ebenfalls einen wertvollen Beitrag leisten.

Betrachtet man das *Sub-Playbook: Vulnerability* fallen keine großen Überraschungen bei einem Blick auf das Prozessdiagramm auf. Die Aufgaben sind im weiteren Sinne alle umgesetzt zu finden und lediglich eingebettet in die restlichen Aufgaben, die bereits im generischen Incident-Handling-Prozess auftreten.

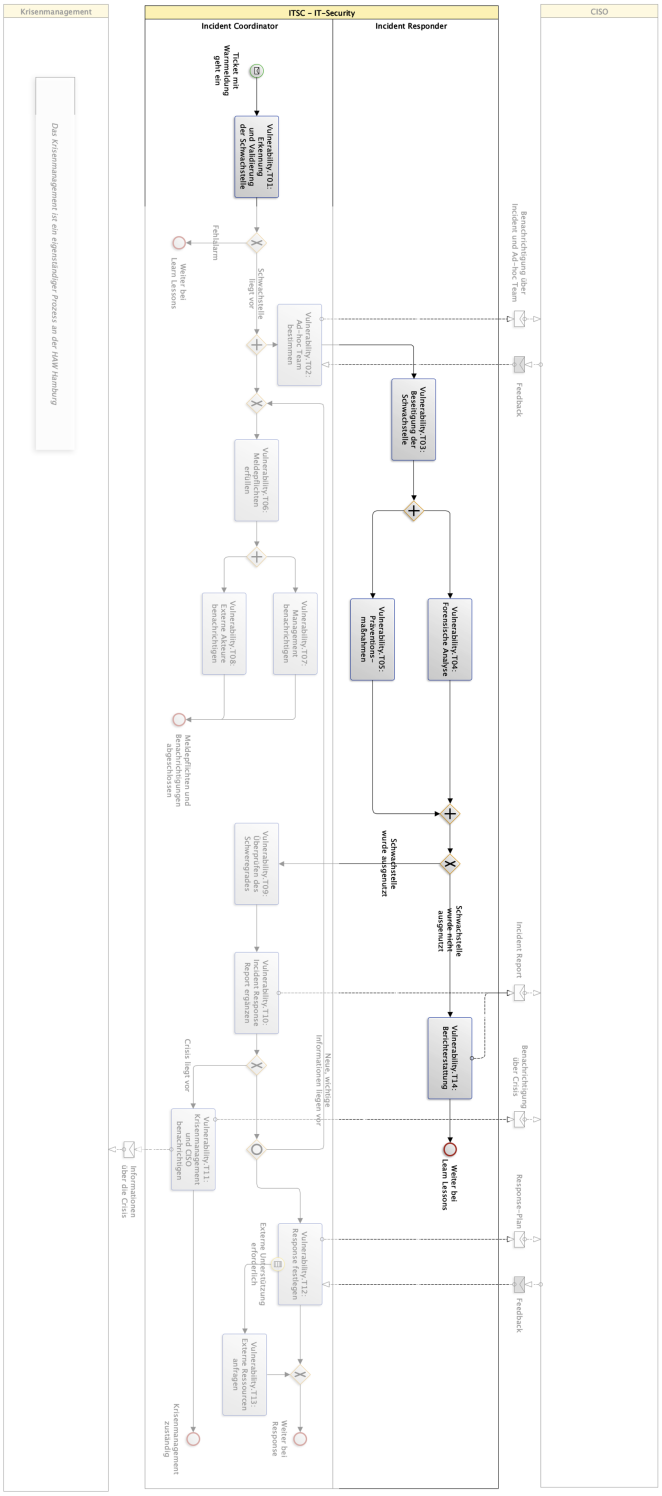

Abbildung 4.5: Prozess: Vulnerability

Security Monitoring/Blocklist

Dieses Prozessdiagramm ähnelt ebenfalls in vielen Punkten dem generischen Prozess wie in Abbildung 4.6 zu sehen ist. Eine Warnmeldung des Meldungstyps *Security Monitoring/Blocklist* dient zwar zunächst auch nur als Hinweis, aber schon hiermit können schwerwiegende Implikationen einhergehen. Für die Aufnahme in eine solche Liste bedarf es verdächtiger Aktivitäten der betroffenen Ressource. Dies könnte lediglich ein Portscanner sein, der als Dienst auf dem System läuft. Es könnte aber auch beispielsweise ein erstes Anzeichen für eine Malware sein, die auf dem System zu finden ist. Hieraus ergibt sich die Notwendigkeit auch für dieses Prozessdiagramm die nötigen Schritte zur Benachrichtigung des Krisenmanagements sowie die Möglichkeit für ein generisches Incident Handling aufzuführen.

In jedem Fall bedarf es zunächst einer Untersuchung der öffentlichen Blocklisten. Wird ein Eintrag in einer solchen Blockliste gefunden, werden zunächst die Ursachen für die Aufnahme untersucht. Sollte sich kein Sicherheitsrisiko dahinter verbergen, gibt es eine Reihe an Möglichkeiten. Zunächst können die Betreiber*innen der Blockliste kontaktiert werden, um den Eintrag entfernen zu lassen. Hat dies keinen Erfolg, gibt es auch noch die Option, das DFN-CERT zu kontaktieren, um Warnmeldungen hierfür auszuschließen. Führt auch dies nicht weiter, kann es in letzter Instanz auch sinnvoll sein, einen Eintrag in einer lokalen Whitelist für die Warnmeldung zu erstellen, damit spätere Warnmeldungen nicht allesamt das komplette Incident Handling triggern (siehe Abschnitt 4.2.2). Solche Whitelists sollten mit besonderer Aufmerksamkeit geführt werden. Wenn beispielsweise die Ressource der Whitelist hinzugefügt wird, könnten spätere Warnmeldungen, die sich auf komplett unterschiedliche Sicherheitsvorfälle beziehen, einfach untergehen. Somit ist es von Vorteil, wenn der Eintrag in der Whitelist möglichst spezifisch ist. Vielleicht sollte beispielsweise nicht nur die IP-Adresse der betroffenen Ressource sondern auch der Meldungstyp eingetragen werden. Je mehr die Warnmeldungen eingegrenzt werden können desto besser. Wie im Prozessdiagramm hinterlegt, ist es zudem unerlässlich, den entsprechenden Eintrag regelmäßig (beispielsweise wie hier vorgeschlagen alle 90 Tage) zu prüfen. Hierdurch wird verhindert, dass bestimmte Einträge einfach untergehen und länger als nötig auf der Whitelist bleiben.

Das zugehörige Sub-Playbook unterscheidet sich im Grunde nicht deutlich von dem tatsächlichen Prozessdiagramm. Es fehlt allerdings hier der Bezug zur Eskalation, die möglich ist, wenn die Warnmeldung Hinweis auf eine größeres Problem gibt. Zudem wird die

Whitelist hier nicht explizit erwähnt. Das Sub-Playbook wurde somit um einige Schritte ergänzt.

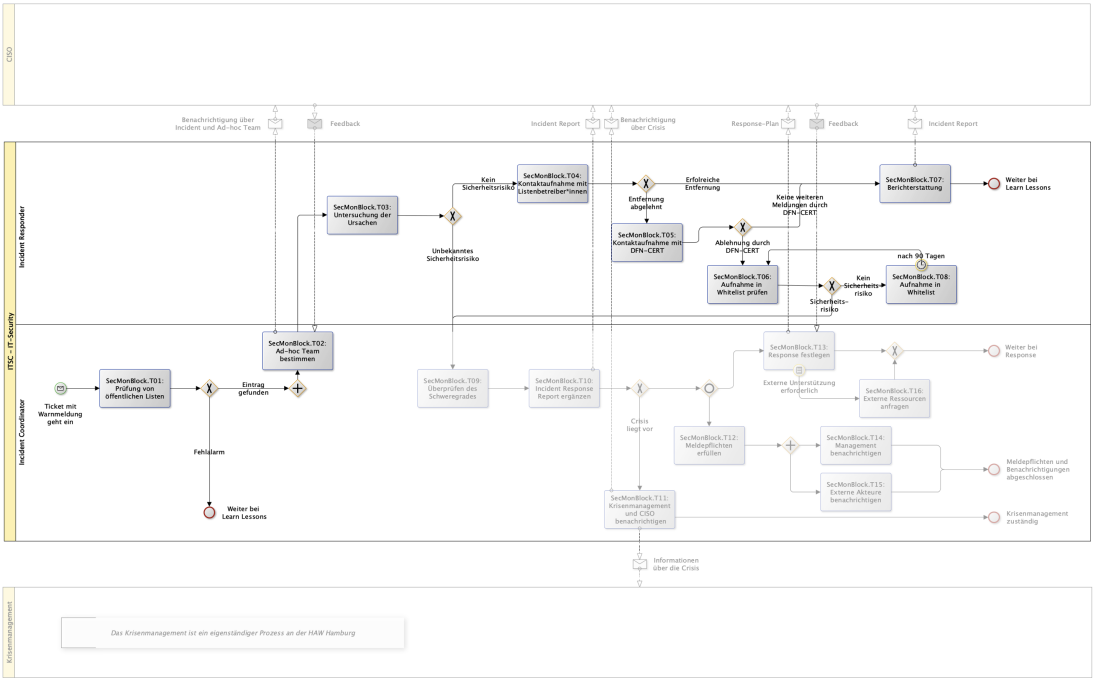

Abbildung 4.6: Prozess: Security Monitoring/Blocklist

Configuration

Zu dem resultierenden Prozessdiagramm des spezifischen Prozesses *Configuration* lässt sich nicht viel Neues sagen. Der Prozess deckt sich fast vollständig mit dem für *Vulnerability*. Entsprechend wurde in Abbildung 4.7 lediglich ein kleiner Ausschnitt aufgeführt, der sich vom Rest der Prozesse unterscheidet. Das komplette Prozessdiagramm ist im Anhang A.5 zu finden. Im Anhang befinden sich auch die restlichen Prozessdiagramme ohne ausgegraute Unterprozesse.

Betrachtet man den Prozessauschnitt in Abbildung 4.7, fällt hier die Unterscheidung zwischen den verschiedenen Unterkategorien des Meldungstyps auf, die für unterschiedliche Warnmeldungen sorgen und unterschiedliche Lösungsansätze mit sich ziehen. So braucht es beispielsweise eine Anpassung des Authentifizierungsmechanismusses bei einer Warnmeldung des Typs 'Configuration/Unrestricted Access'. Ist keine Unterkategorie angegeben, fällt die entsprechende Aufgabe entsprechend generischer aus.

Das *Sub-Playbook: Configuration* deckt sich zwar in vielerlei Hinsicht mit den aufgeführten Aufgaben im resultierenden Prozessdiagramm, dennoch gibt es einige Unterschiede, die es anzumerken gilt:

Zunächst gibt es keine wirkliche Notwendigkeit für eigene Prozessdiagramme pro Unterkategorie. Auch wenn die ursprüngliche Strukturierung mit Sub-Sub-Playbooks dies nahelegt, unterscheiden sich die Prozesse letztlich in zu geringem Maße, um dies zu rechtfertigen. Besonders aufgefallen ist hier die Dopplung in den Aufgaben 'Erkennung und Validierung der unerwünschten Konfiguration'. Diese wurde zwar in den Sub-Sub-Playbooks leicht anders benannt, letztlich umfasst dies jedoch ähnliche Aufgaben, die in einer Aufgabe zusammengefasst werden können. Zu einem späteren Zeitpunkt könnte es auch hier sinnvoll sein, die Prozessdiagramme nach den jeweiligen Unterkategorien aufzuspalten; für den jetzigen Zeitpunkt gibt es dazu jedoch keinen Anlass.

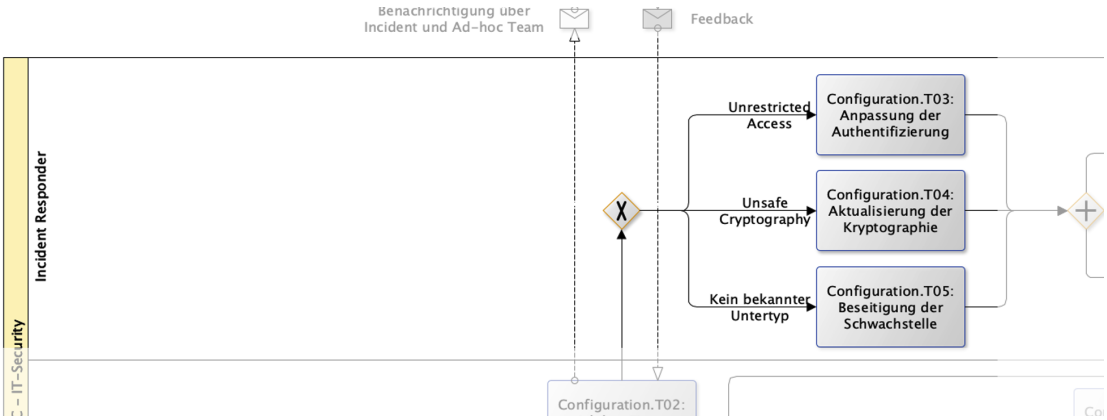

Abbildung 4.7: Ausschnitt Prozess: Configuration

Einordnung der Prozessdiagramme

Die an dieser Stelle erarbeiteten Prozessdiagramme dienen hier dazu, einen Bogen zu bilden zu der Arbeit in [12], bei der der generische Incident-Handling-Prozess als BPMN-Diagramm erarbeitet wurde. Entsprechend wurde sich an dieser Notation und Darstellungsform orientiert, um sie um Unterprozesse zu erweitern, die auf den Meldungstypen basieren, die nach der Anwendung der Filter bei den restlichen Warnmeldungen auftreten. Nichtsdestotrotz ist dies lediglich ein erster Schritt zur Beschreibung dieser Prozesse. Wie im Grundlagen-Kapitel bereits beschrieben wird, verwenden nur wenige Automatisierungslösungen tatsächlich BPMN, um ihre Playbooks zu definieren. Die

Wiederverwendbarkeit über eine Veranschaulichung hinaus ist bei diesem Format somit fragwürdig.

Da Ziel dieser Arbeit vordergründig eine einheitliche Definition der Incident-Handling-Prozesse ist, die um die spezifischen Prozesse erweitert werden sollten, wurde diese Erweiterung in den Vordergrund gestellt und die Verwendung für Implementierungsmöglichkeiten im Rahmen dieser Arbeit weniger berücksichtigt.

5 Von der Theorie zur Praxis

Der Großteil dieser Arbeit beschäftigte sich bisher damit, einen theoretischen Unterbau zu schaffen, der dann als Grundlage für eine Optimierung von Prozessen dienen kann. So wurden nun eine Vielzahl an Daten analysiert, es wurden Filterkonzepte erstellt und Prozessdiagramme verfasst. All das scheint auf dem Papier fern von der Praxis zu sein. Wie unmittelbar die Implikationen für den praktischen Nutzen sind, geht dabei schnell unter. Das folgende Kapitel versucht dies aufzubrechen, indem es ein stärkeres Licht auf genau diesen Bezug zwischen der bisherigen, theoretischen Betrachtung und dem praktischen Mehrwert wirft.

Es geht allerdings nicht nur um den praktischen Mehrwert dieser Arbeit im Speziellen. Stattdessen wird darüber hinaus auch aufgeführt, wie dieser Nutzen in der Zukunft noch stärker herausgearbeitet werden kann und was für Schritte an diese Arbeit anschließen sollten.

5.1 Auswirkungen

Die Resultate der Bearbeitung der Daten von dieser Arbeit wurden bereits in Abschnitt 3.3 näher beleuchtet. Dieser Abschnitt betrachtete vordergründig die resultierenden Daten selbst und weniger die Auswirkungen, die aus den Maßnahmen dieser Arbeit resultieren.

Die exakten Auswirkungen, die diese Arbeit auf den Incident-Handling-Prozess der HAW Hamburg hat, lassen sich vermutlich nur durch eine weitere Anschlussarbeit und praktische Experimente fassen. Es wäre somit anmaßend, davon auszugehen, in diesem Kapitel ein exaktes Bild davon zeichnen zu können. Eine grobe Skizze zu zeichnen, lohnt sich aber, um den Mehrwert dieser Arbeit besser greifen zu können und insbesondere Anreize dafür zu schaffen, die theoretischen Ergebnisse dieser Arbeit in der Praxis umzusetzen.

Die finanziellen Mittel an der HAW Hamburg sind wie bei fast allen Hochschulen in Deutschland eine knappe Ressource. Die interessanteste Frage, die sich in Hinblick auf die Implikationen dieser Arbeit stellt, ist somit, was für Einsparungsmöglichkeiten sich aus den Ergebnissen der Arbeit ergeben. Um eine grobe Schätzung für diese Einsparungsmöglichkeiten zu geben, bedarf es zunächst einer Art Heuristik.

Eine Heuristik dafür aufzusetzen, wie viel Zeit bei der Bearbeitung gespart wird, wenn ein spezifisches Playbook statt des generischen Incident-Handling-Prozesses genutzt wird, kann recht willkürlich sein und stark zwischen den verschiedenen Sicherheitsvorfällen variieren. Zwar gibt es definitiv Einsparungen, sollten Prozesse klarer definiert sein, weil hierdurch keine Gedanken daran verschwendet werden müssen, wie das genaue Vorgehen ist, es könnte aber auch sein, dass die Person, die den Vorfall bearbeitet bereits 100 ähnlicher Incidents behandelt hat und insofern genau über das Vorgehen Bescheid weiß. Dementsprechend wird bei der nachfolgenden Berechnung die tatsächliche Response außen vor gelassen. Stattdessen wird sich darauf fokussiert, wie der aktuelle Prozess aussieht, bei dem entschieden wird, ob es sich bei einer Warnmeldung um einen potenziellen Sicherheitsvorfall handelt oder nicht.

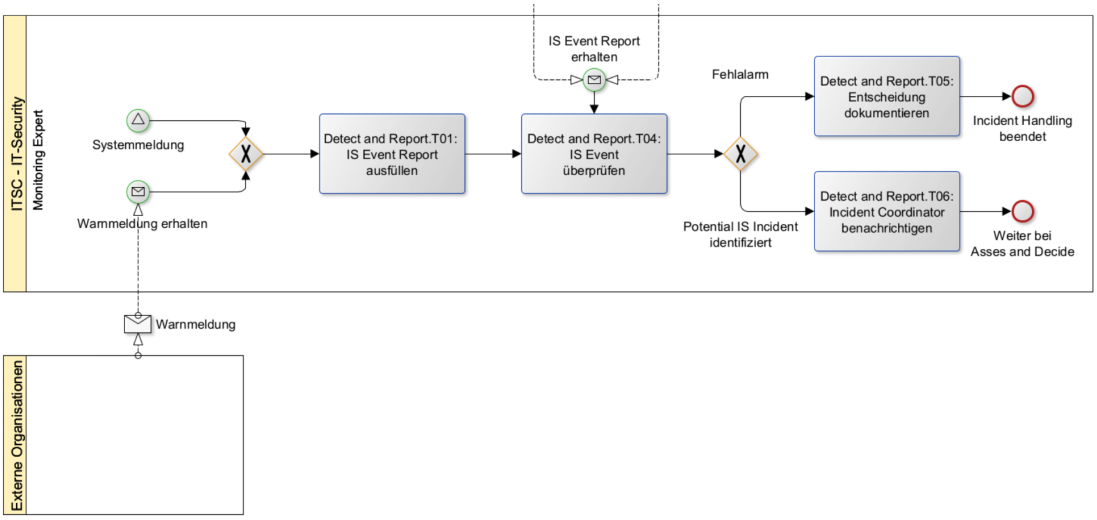

Abbildung 5.1: Ursprünglicher Prozess zur Bearbeitung von Warnmeldungen - aus [12]

Der aktuelle Prozess hierzu ist in Abbildung 5.1 zu sehen. Hierbei handelt es sich um einen Ausschnitt aus dem Prozessdiagramm zur Phase *Detect and Report* aus [12]. Hierbei fallen einige Dinge ins Auge. Zunächst wird dieser Teil der Entscheidungsfindung von einer einzelnen Person, dem *Monitoring Expert* übernommen. Wenn eine Warnmeldung eingeht, muss zunächst ein IS Event Report ausgefüllt werden, im Anschluss daran folgt

eine Prüfung der Warnmeldung. Bei dieser Prüfung könnte somit herauskommen, dass es bereits eine Bearbeitung des Vorfalls gibt, weil es sich dabei um eine doppelte Meldung handelt oder vergleichend um eine Wiederholungsmeldung. Ein anderes Beispiel für einen Fehlalarm wäre es, wenn der Monitoring Expert feststellt, dass die Warnmeldung durch einen bekannten Dienst wie einen der Portscanner ausgelöst wurde. Sollte sich für eine Fehlmeldung entschieden werden, muss anschließend diese Entscheidung noch dokumentiert werden. Betrachtet man nur die Warnmeldungen, bei denen es sich um Fehlmeldungen handelt, könnte man eine Bearbeitungsdauer von 15 Minuten ansetzen - also für das Erstellen eines Reports, der Prüfung der Warnmeldung und der Dokumentation der Entscheidung. In den 228 Tagen der betrachteten Daten sind insgesamt 2.114 Warnmeldungen eingegangen, wovon lediglich 92 AW-Meldungen gemeldet wurden, für die der Incident-Handling-Prozess nach dieser Phase nicht direkt hätte beendet werden können.

5.1.1 Berechnung

Um die tatsächlichen Ressourcen zu berechnen, die für diesen Prozess aktuell aufgewendet werden, muss ein Stundensatz für die Bearbeitung festgelegt werden. Für diese Arbeit wurde sich hierbei für den Stundensatz von *Studentischen Hilfskräften* an der HAW Hamburg entschieden. Dieser Satz liegt Stand November 2024 bei 13,25€ pro Stunde [10]. Der Begriff *Monitoring Expert* deutet zwar zunächst auf etwas anderes hin, aber für die Ausübung dieser Tätigkeit dürfte die Expertise einer studentischen Hilfskraft ausreichen. Zumal die tatsächliche Triage erst in der Phase *Assess and Decide* angegangen wird und an diesem Punkt lediglich die Aussortierung von Fehlmeldungen im Fokus steht.

Gegebene Werte:

- Stundenlohn einer studentischen Hilfskraft: 13,25 € pro Stunde
- Zeitraum der Warnmeldungen dieser Arbeit: 228 Tage
- Gesamtanzahl der Warnmeldungen: 2114 Warnmeldungen
- Anzahl der nicht filterbaren Warnmeldungen: 92 Warnmeldungen
- Bearbeitungszeit pro Warnmeldung: 15 Minuten

Berechnung der gefilterten Warnmeldungen pro Jahr:
In den 228 Tagen an Datensätzen, die bei dieser Arbeit betrachtet wurden, gingen 2114 AW-Meldungen ein. Die Anzahl der Warnmeldungen, die nach dieser Arbeit eindeutig hätten gefiltert werden können, lautet:

$$\text{Gefilterte Warnmeldungen} = 2114 - 92 = 2022$$

Für diese Berechnung wird geschaut, wie viel die HAW Hamburg pro Jahr an Einsparungsmöglichkeiten hierfür hätte. Insofern wird nun die Anzahl an Warnmeldungen hochgerechnet, die pro Jahr gefiltert werden können:

$$\text{Gefilterte Warnmeldungen pro Tag} = \frac{2022}{228} \approx 8,87$$

Pro Jahr ergeben sich somit in etwa:

$$\text{Gefilterte Warnmeldungen pro Jahr} = 8,87 \times 365 \approx 3234$$

Berechnung des Ressourcenaufwands:
Da die Bearbeitungszeit pro gefilterter Warnmeldung 15 Minuten beträgt, berechnet sich die Gesamtzeit, die pro Jahr hierfür aufgewendet wird, so:

$$\text{Bearbeitungszeit pro Jahr} = 3234 \times 15\,\text{Minuten} = 48510\,\text{Minuten}$$

In Stunden umgerechnet somit:

$$\text{Bearbeitungszeit pro Jahr in Stunden} = \frac{48510}{60} \approx 808,5\,\text{Stunden}$$

Hieraus lassen sich entsprechend die finanziellen Kosten berechnen, die mit diesem Prozess einhergehen:

$$\text{Kosten pro Jahr} = 808,5\,\text{Stunden} \times 13,25\,€ \approx 10713,63\,€$$

5.1.2 Ergebnis

Die geschätzten Kosten, die durch den Prozess jährlich anfallen sind mit 10713,63€ immens. Allerdings gilt noch einmal zu betonen, dass es sich hierbei um eine sehr vage Schätzung handelt. Der tatsächliche Wert könnte hiervon stark abweichen. Bei der obigen

Rechnung wird davon ausgegangen, dass tatsächlich jede Warnmeldung eingehend geprüft wird. Die 15 Minuten, die für jede Warnmeldung aufgewendet werden müssen, könnten zu hoch angesetzt sein. Wenn sich ein *Monitoring Expert* regelmäßig mit diesen Meldungen befasst, ist es möglich, dass die Person nach einer Weile nur einen Blick braucht, um einige der Fehlmeldungen zu identifizieren. Damit könnte die Summe im Endeffekt deutlich niedriger ausfallen.

Allerdings gibt es auch diverse Faktoren, die dafür sorgen könnten, dass die geschätzten Kosten entschieden höher ausfallen. Die 15 Minuten könnten beispielsweise für manche Meldungen deutlich zu kurz sein. So könnte es einiges an Zeit beanspruchen, herauszufinden, dass bereits vor zwei Monaten für eine bestimmte Schwachstelle ein Incident-Handling-Prozess gestartet wurde. Zudem sind in der obigen Rechnung lediglich die Fälle aufgeführt, bei denen die Warnmeldungen als irrelevant eingestuft werden können. Auch die relevanten Meldungen (hier 92) brauchen ebenfalls Zeit, um geprüft zu werden, selbst wenn die Entscheidung, laut Prozessdiagramm in 5.1, nicht gesondert dokumentiert werden muss.

Der größte Faktor, der hier aber in dieser Rechnung nicht betrachtet wird, ist, dass der Stundensatz von 13,25€ sehr niedrig gehalten ist. In der Realität könnte dieser Satz um ein Vielfaches höher ausfallen als dieser Wert, der nur knapp über dem Mindestlohn liegt. Zudem sind hier weder mögliche weitere Kosten wie Sozialversicherung noch Kosten für Einstellung, Onboarding oder auch Ausfälle durch Krankheiten oder Urlaube mit einbezogen. Alles in allem kann eine Einsparungsmöglichkeit von über 10.000 Euro pro Jahr also als durchaus realistisch betrachtet werden.

Wie hoch die Summe auch exakt ist, dass die Kosten durch die hier vorgeschlagenen Filter erheblich reduziert werden würden, steht außer Frage. Die Implementierung einer Automatisierung mit diesen Filtern wäre ein erster Schritt für ein immenses Potenzial, dass sich hinter der Automatisierung weiter Teile des Incident-Handlings an der HAW Hamburg versteckt.

5.2 Praktische Umsetzung der Automatisierung

In den vorherigen Kapiteln wurde mehrfach darüber gesprochen, dass mit dieser Arbeit ein Grundstein für die Automatisierung der Incident-Response- und generell der Incident-Handling-Prozesse an der HAW Hamburg gelegt wird. Es wurde dargestellt,

welche Vorteile sich aus einer solchen Automatisierung ergeben und es wurde detailliert beschrieben, wie der aktuelle Incident-Handling-Prozess der Hochschule aufgebaut ist.

Was an dieser Stelle allerdings noch fehlt, sind konkrete Aufgaben, die sich an diese Grundlagenarbeit anschließen. Diese Arbeit zielt darauf ab, ein praktisches Beispiel aufzuzeigen, wie Incident-Response-Prozesse Schritt für Schritt automatisiert werden können. Es sollte jedoch auffallen, dass zum Stand nach dieser Arbeit noch keine echte Automatisierung erreicht wurde. Somit fehlen zumindest konzeptuelle Anweisungen, die auf die hier beschriebenen Ansätze aufbauen.

Somit stellt sich die Frage: *Was folgt auf diese Arbeit?*

1. **Technische Infrastruktur schaffen:** Zuallererst ist es notwendig, sich mit der nötigen, technischen Infrastruktur zu beschäftigen. Hieraus ergeben sich die folgenden Teilaufgaben

   - *Betrachtung der Systeme für die Incident-Response-Prozesse der HAW Hamburg:* Hier müssen die bestehenden Systeme analysiert werden. Es ergibt wenig Sinn, an dieser Stelle die Einführung eines neuen SIEM-Tools vorzuschlagen, wenn bereits ein anderes SIEM-Tool existiert und genutzt wird.

   - *Notwendige Tools bestimmen:* In diesem Schritt werden die konkreten Tools und Technologien bestimmt, die für die Automatisierung der Incident-Response-Prozesse in den aktuellen Systemen noch fehlen. Ein SIEM-Tool wäre beispielsweise tatsächlich sinnvoll, schon um Security Events für die eingehenden Warnmeldungen zu erstellen. Auch ein SOAR-Tool wäre essenziell, um die verschiedenen Aufgaben des Incident-Handling zu orchestrieren. In diesem Schritt sollte sich nicht nur für generelle Technologien sondern schon für spezifische Implementierungen entschieden werden. Es gibt beispielsweise eine Reihe an SOAR-Lösungen. Welche genau für die jeweiligen Systeme am sinnvollsten wäre, muss spezifisch recherchiert und abgewogen werden.

   - *Aufsetzen der Tools:* Nachdem eine Liste der notwendigen Tools besteht, müssen diese noch aufgesetzt werden.

2. **Teilautomatisierung:** Bestimmte Aufgaben, die automatisiert werden müssen, sind sehr klar. Hierzu zählt beispielsweise die Automatisierung der in dieser Arbeit vorgeschlagenen Filter. Es wäre möglich, eine eingehende AW-Meldung zu erfassen, diese auf die verschiedenen Filterkriterien zu prüfen, eventuell auszusortieren und

für die relevanten Meldungen sogar direkt ein Ticket für die weitere Bearbeitung zu erstellen, ohne dass eine einzige manuelle Handlung nötig wäre.

3. **Verfeinerung der Playbooks und Wege zur Automatisierung:** Für die weitgehende Automatisierung der spezifischen Playbooks sind diese noch zu generisch gehalten. Die darin beschriebenen Aufgaben sind zwar grundlegend korrekt, aber erfassen sie sehr allgemein für beliebige CSIRTs. Dadurch sind sie definitiv nicht ausreichend auf die HAW Hamburg zugeschnitten, um eine weitgehende Automatisierung zu ermöglichen. Hierfür müsste mit den entsprechenden Teams direkt in Kontakt getreten und die aktuellen Prozesse genauer untersucht werden. Insbesondere wäre es hierfür essenziell, die aktuell genutzten Tools und deren Schnittstellen zu erfassen, um diese womöglich direkt in die Orchestrierung durch das SOAR-Tool einzubeziehen oder Alternativen zu suchen, für die dies eine Möglichkeit wäre. An dieser Stelle wäre es eventuell auch sinnvoll, einen Blick auf die Arbeit in [16] zu werfen, in der verschiedene Tool-Arten für Incident-Response-Prozesse vorgestellt werden.

Auch wenn versucht wurde, die Frage 'Was folgt auf diese Arbeit?' mit einer Liste zu beantworten, sollte an dieser Stelle betont werden, dass es sich hierbei um keine strikte Abarbeitung von Aufgaben handelt, die nacheinander abgearbeitet werden können und nach Erfüllung aller Aufgaben ist die Automatisierung erfolgreich abgeschlossen. Tatsächlich sind gerade die letzten beiden Einträge der Liste vielmehr als ein Kreislauf gedacht, indem es eine stetige Verfeinerung der Playbooks und eine darauf basierende weitere Automatisierung geben wird. Mit neuen Aufgaben wird es zudem auch immer neue Notwendigkeiten für Automatisierungen geben. Es handelt sich somit vielmehr um einen laufenden Prozess, für den mit dieser Arbeit versucht wird, die Tür weiter aufzustoßen, als dass es einen abgeschlossenen Rahmen geben wird, in dem diese Automatisierung behandelt werden kann.

5.3 Erster Implementierungsvorschlag

Die Idee dieses Kapitels ist es, die praktische Anwendung der theoretischen Betrachtung dieser Arbeit in den Vordergrund zu stellen. Der vorherige Abschnitt beschreibt die *Praktische Umsetzung der Automatisierung*. Das Ganze bleibt allerdings sehr abstrakt und ist somit alleinstehend noch weit weg von der Praxis, die hier im Fokus stehen soll.

Ein Teil dieses Praxisbezugs ist es, den Realitätsbezug nicht zu verlieren. Ein Fokus dieser Arbeit ist die Automatisierung und wie sich durch Automatisierungstools mit den Playbooks als Grundlage erhebliche Optimierungsmöglichkeiten für Incident-Response-Prozesse ergeben. Der deutliche Mehrwert der Arbeit liegt aber unmittelbar in den Filtern, die hier erarbeitet wurden. Und wenn an dieser Stelle der theoretische Bezug verlassen und ein schärferer Blick auf die Praxis geworfen wird, fällt schnell auf, dass es für die Umsetzung dieser Filter im Grunde keine komplexen Automatisierungstools braucht. Um von den Vorteilen der Filtermöglichkeiten zu profitieren, ist eine simple Architektur vollkommen ausreichend. Diese Architektur wird im Folgenden vorgestellt.

5.3.1 Architektur

Abbildung 5.2 zeigt eine simple Architektur, die bereits ausreichend wäre, um automatisiert von den Filtermöglichkeiten profitieren zu können.

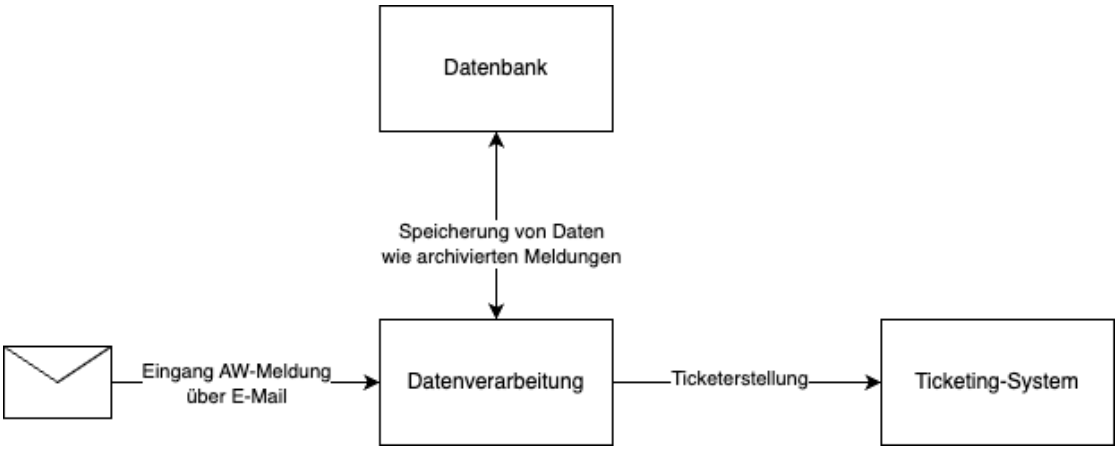

Abbildung 5.2: Möglicher Aufbau der Implementierung

Beschreibung der Dienste:

- *Datenverarbeitung:* Der Dienst der Datenverarbeitung steht bewusst im Zentrum des Diagramms. Dieser Dienst greift zunächst auf die eingehenden Warnmeldungen zu. Diese gehen täglich um 10 Uhr und 14 Uhr per E-Mail ein und der E-Mail-Anhang enthält eine XML-Darstellung aller Warnmeldungen, die seit der letzten E-Mail beim DFN-CERT erstellt wurden. Im Anhang in A.2 findet sich das Schema der XML-Dateien. Somit sind die Warnmeldungen maschinenlesbar und der Dienst kann an dieser Stelle automatisiert auf eingehende Warnmeldungen zugreifen. Eine

API ist seit Sommer 2024 ebenfalls in Entwicklung. Der Fortschritt dieser Entwicklung sollte erneut in Erfahrung betrachtet werden, bevor mit der Anbindung per E-Mail begonnen wird, da dies vermutlich eine leichtere und stabilere Einbettung in den Datenverarbeitungs-Dienst ermöglichen würde.

Im Datenverarbeitungs-Dienst werden des Weiteren die verschiedenen Filterschritte durchlaufen, die in Kapitel 3 vorgestellt wurden. Für einige dieser Filter ist es notwendig, Daten zu persistieren. Beispielsweise können *doppelte Meldungen* nur erkannt werden, wenn die bisher eingegangenen Meldungen gespeichert werden. Hierzu dient die aufgeführte Datenbank.

Der Dienst könnte auch um weitere Funktionalitäten erweitert werden. So könnte auch eine Datenanreicherung eingebaut werden. Dazu würden aus externen Quellen Informationen abgerufen werden - beispielsweise zusätzliche Angaben über eine Schwachstelle basierend auf der CVE-Nummer. Diese könnten dann bei der Ticketerstellung ebenfalls direkt dem Ticket beigefügt werden. Da es hier allerdings um einen ersten Implementierungsvorschlag geht, wurde dies nicht in Abbildung 5.2 aufgeführt.

Nach der Verarbeitung der Warnmeldungen kann nun ein Ticket im Ticketing-System erstellt werden. Die gesamte Implementierung des Datenverarbeitung-Dienstes ist hier sehr generisch beschrieben. Grund hierfür ist, dass diese Implementierung vermutlich in jeder gängigen Backend-Programmiersprache unter Verwendung einer noch größeren Vielfalt an Frameworks möglich ist. Entsprechend vage wird sich auch hier in der Beschreibung verhalten.

Der Datenverarbeitungs-Dienst könnte auch in mehrere Microservices aufgeteilt werden. Allerdings gilt zu beachten, dass die Kommunikationsschnittstellen zwischen diesen Diensten sowie die damit verbundenen Sicherheitsanforderungen eine zusätzliche Komplexität in die Architektur bringen. Da hier von einem simplen Implementierungsansatz ausgegangen wird, wurde sich an dieser Stelle für eine Monolith-Architektur entschieden.

- *Ticketing-System:* Auch für das Ticketing-System gibt es eine Reihe an Möglichkeiten. Sollte für die aktuelle Bearbeitung der Aufgaben bereits ein Ticketing-System bestehen, wäre es sinnvoll, dieses im Zuge dieser Implementierung weiter zu nutzen und einfach anzubinden. Ist dies nicht der Fall, wäre es sinnvoll, eine bestehende Lösung im System aufzusetzen. Hier gibt es eine Reihe an Optionen, auf die

im Nachgang näher eingegangen wird. Basierend auf dieser Wahl entscheidet sich auch die Kommunikation zwischen der Komponente zur Datenverarbeitung und dem Ticketing-System.

5.3.2 Wahl des Ticketing-Systems

Wenig ist so vielen Änderungen ausgesetzt, wie die beste Wahl eines Software-Tools für eine bestimmte Aufgabe. Schon zu dem Zeitpunkt der Veröffentlichung dieser Arbeit könnte diese Wahl somit obsolet sein. Um möglichst nah an der Praxis zu bleiben, wird nun dennoch ein Vorschlag für ein mögliches Tool gemacht, auch um beispielhaft aufzuzeigen, welche Kriterien hier relevant sind. Es gilt allerdings zu beachten, dass eine weitere Recherche für den Fall der tatsächlichen Implementierung deutlich angeraten wird.

Da an dieser Stelle ein neues Tool vorgeschlagen wird, was primär zum Tracken des Fortschritts der Bearbeitung von AW-Meldungen dienen soll, ist die Liste der minimalen Anforderungen recht überschaubar. Die erste Anforderung wäre, dass sich Tickets erstellen lassen, die den Fortschritt der Bearbeitung von Warnmeldungen verfolgen können. Es muss zudem möglich sein, diese Tickets automatisch über eine Schnittstelle zu erstellen, an die der Dienst *Datenverarbeitung* angebunden werden kann. Weitere Anforderugen wären eine bedienbare Oberfläche und eine Userverwaltung, um die verschiedenen Personen, die die Warnmeldungen bearbeiten sollen, einzelnen Tickets zuzuweisen. Eine Priorisierungsmöglichkeit für Tickets wäre ebenfalls sinnvoll, um die Prioritäten, die sich aus dem Schritt der Prioritätsbewertung ergeben, im Ticket abbilden zu können.

Was an dieser Stelle an funktionalen Anforderungen beschrieben wird, wird im Prinzip von jedem Ticketing-System erfüllt, das es auf dem Markt gibt. Dies kommt in diesem Fall zu Gute, da somit auch ohne Weiteres Open-Source-Tools in Frage kommen und nicht auf proprietäre Software wie Jira mit kostenpflichtigen Angeboten zurückgegriffen werden muss. Die größte Open-Source-Alternative stellt *OpenProject* dar [21]. Auf ihrer Website beschreiben sie die Community Edition, die eine komplette Open-Source-Lösung darstellt und ohne Weiteres auf Linux oder mit Hilfe von Docker in der Umgebung aufgesetzt werden kann. Die Lösung bietet alle notwendigen Features, die oben beschrieben wurden und über dies hinaus auch zahlreiche weitere Möglichkeiten. So wird als Schnittstelle nicht nur eine normale REST-Schnittstelle angeboten sondern HATEOS (Hypermedia as the Engine of Application State), was eine Erweiterung (oder streng genommen ein Constraint) von REST darstellt. Durch die Bereitstellung von *Hypermedia*-Links in

der Response kann der Client (hier der Datenverarbeitungs-Dienst) dynamisch durch Ressourcen navigieren. Es braucht somit zum Zeitpunkt der Implementierung kein umfassendes Wissen über die bereitgestellten Endpunkte und es folgt eine lose Kopplung [11]. Die Authentifizierungsmöglichkeiten sind ebenfalls vielfältig. Neben der Authentifizierung mittels API-Key wird beispielsweise auch OAuth2.0 oder direkt OIDC angeboten. Lediglich Single Sign-On wird nur in der Enterprise Edition ermöglicht.

Ein Beispiel für ein Ticket in OpenProject kann in Abbildung 5.3 betrachtet werden. Hier zeigen sich auch weitere der besprochenen Anforderungen. Beispielsweise gibt es ein Feld für die Priorität des Tickets und das Ticket kann einer konkreten Person zugewiesen werden.

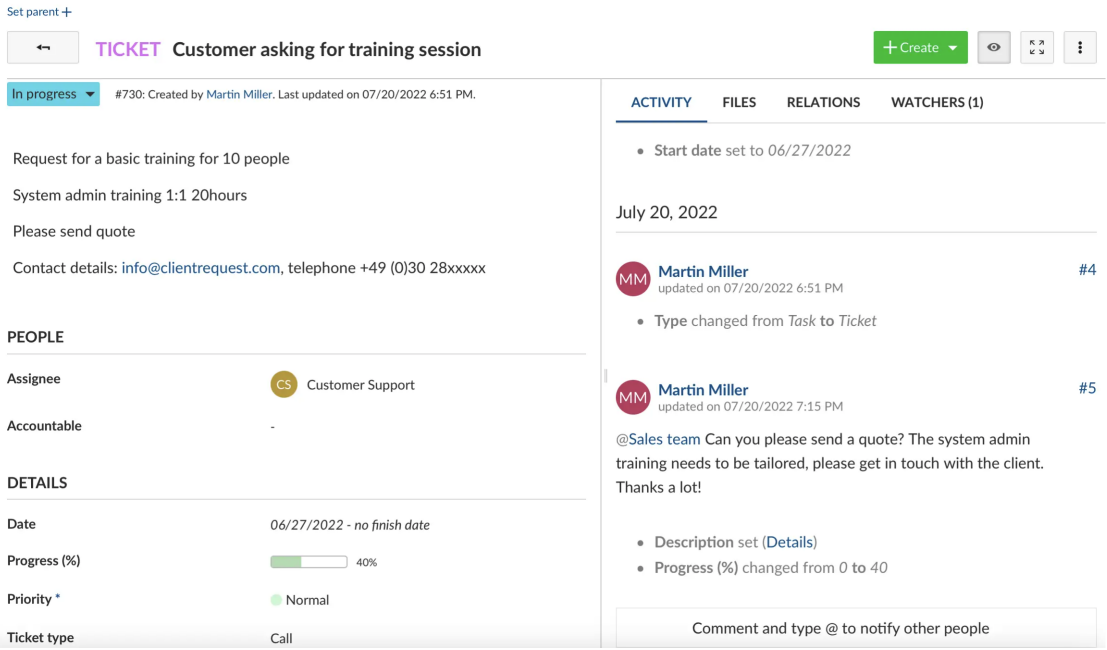

Abbildung 5.3: Beispiel für ein Ticket in OpenProject - aus [4]

5.3.3 Nachtrag zum Implementierungsvorschlag

Aus diesem Abschnitt des Implementierungsvorschlags ergibt sich ein gewisser Zwiespalt für diese Arbeit. Zum einen sollte definitiv sichergestellt werden, dass die hier vorgestellten Filtermöglichkeiten genutzt werden. Zum anderen stellt diese Implementierung auch ein gewisses Risiko für die finale Lösung dar. Denn sobald diese erste Implementierung erst einmal aufgesetzt ist, ist der Pain Point im Anschluss entsprechend marginal, da der

Prozess deutlich optimiert und bereits teilweise automatisiert wurde. Somit schrumpft die Wahrscheinlichkeit dafür, dass jemals eine umfassende Automatisierung des Incident-Response-Prozesses an der HAW Hamburg umgesetzt wird.

Nichtsdestotrotz wird dieser Implementierungsansatz hier beschrieben, um ein Licht darauf zu werfen, dass schon mit verhältnismäßig kleinem Aufwand eine Transformation von Incident-Response-Prozessen durch Automatisierungen möglich ist. Es ist wichtig, die Automatisierung dieser Prozesse nicht als etwas zu verstehen, bei dem es einen klaren Weg gibt, der beschritten werden sollte. Vielmehr ist es oft ein Ausloten verschiedener Möglichkeiten und Tools und es ist entscheidend hier immer auch zwischen Kosten und Nutzen abzuwägen. Auch bei dem in dieser und vergangener Arbeiten beschriebenen SOAR-Ansatz verhält es sich ähnlich, denn diese Lösung ist nicht die Lösung für alles. Hierauf wird im folgenden Abschnitt näher eingegangen.

5.4 Kritik an SOAR-Tools

Wie der Titel der Arbeit verrät, beschäftigt sich diese Arbeit mit der Automatisierung von Incident-Response-Prozessen. Beim Lesen der übrigen Arbeiten, die dieser hier zu Grunde liegen ([14], [15] und [16]) werden Automatisierung und SOAR-Tools ohne weitere Differenzierung fast gleichgesetzt. Auch in der Liste der weiteren Umsetzungsmöglichkeiten in Abschnitt 5.2 wird direkt ein SOAR-Tool als zentrale Stelle für die Automatisierung vorgeschlagen. Es könnte jedoch sinnvoll sein, die Begriffe der Automatisierung von Incident Response und SOAR-Lösungen weiter voneinander zu trennen.

Der Begriff SOAR sorgt in den Köpfen von vielen Expert*innen in dem Bereich Cyber Security mittlerweile von Kopfschütteln bis hin zu Kopfschmerzen. Dies geht aus einem neuen Report von Gartner aus diesem Sommer hervor [24]. Der Grund hierfür ist, dass der theoretische Nutzen in der Praxis meist schwer zu erreichen ist. Wie schon bei der HAW Hamburg mit ihren starren Incident-Handling-Prozessen sieht die Realität der zu automatisierenden Prozesse meist komplex aus, bei denen sich kein klarer Weg abzeichnet, der beschritten werden kann, um eine Automatisierung der Prozesse zu erreichen. Dadurch ergeben sich eine Vielzahl an Hürden. Zunächst ist der initiale Kostenaufwand immens. Die vorliegende Arbeit kann als Beispiel dazu dienen, wie aufwendig es bereits ist, eine Beschreibung der Prozesse und erste Optimierungsmöglichkeiten anzugehen. Um auch nur eine Teilautomatisierung zu erreichen, bedarf es zunächst einer intensiven Untersuchung der bestehenden Systeme und einer genauen Abwägung der einzusetzenden Tools,

bevor mit der Transformation der Prozesse überhaupt begonnen werden kann. Auch nach den initialen Kosten braucht es fortlaufende Betreuung von geschultem Personal und die tatsächliche Intergration und Interoperabilität von verschiedenen Diensten ist nicht immer möglich. Besonders fragwürdig sei jedoch das Versprechen von SOAR-Tools, die eine zentrale Sicherheitslösung darzustellen, die alles in sich vereint und jedwede Probleme lösen kann. Wie schon zuvor beschrieben, ist es ein langer Weg hin zur Automatisierung von Prozessen und somit wären schon Teilautomatisierungen an der HAW Hamburg echte Erfolge.

Dies ist jedoch keine Empfehlung für Resignation oder gar nichts weiter zu unternehmen - im Gegenteil. Wie ebenfalls in [24] beschrieben wird, werden zwar SOAR-Tools selbst dem Hype nicht gerecht, die Automatisierung ist hingegen ein notwendiger Wandel für eine weitreichende Optimierung von Incident Response. Andere Automatisierungs-Tools, die stärker auf Künstliche Intelligenz setzen, seien aktuell im Kommen. Die Automatisierung ist in jedem Fall allgegenwärtig. Ein Beispiel für die Optimierungsmöglichkeiten durch Automatisierung stellt schon diese Arbeit dar. Die angeführte Heuristik in Abschnitt 5.1 stellt allein durch die Entwicklung von automatisierten Filtern eine jährliche Einsparung der HAW Hamburg im fünfstelligen Bereich in Aussicht. SOAR-Tools selbst mögen somit keine tautologische Besonderheit darstellen, Schritte in Richtung Automatisierung scheinen jedoch nach wie vor unumstritten zu sein. Die spezifischen Tool-Möglichkeiten sollten dann im jeweiligen Einzelfall genau abgewogen werden.

6 Fazit

Nachdem im vorherigen Kapitel ein Bogen zur Praxis geschlagen wurde, gibt das nachfolgende Kapitel nun noch einmal ein umfassendes Fazit darüber, was genau erarbeitet wurde und mit welchen Limitationen diese Arbeit einhergeht. Zudem gibt es einen Ausblick auf die Arbeiten, die an diese Arbeit anknüpfen können und welchen weiteren Herausforderungen die Automatisierung der Incident-Response-Prozesse der HAW Hamburg gegenübersteht.

6.1 Erkenntnisse

Die grundlegende Aufgabe, die im Fokus dieser Arbeit stand, war es, darzustellen, wie Incident-Response-Prozesse automatisiert werden können. Dies sollte nicht rein theoretisch aufgearbeitet werden, weswegen das Beispiel der HAW Hamburg und die Reaktion auf automatische Warnmeldungen zentral betrachtet wurden.

Nach einer Definition von grundlegenden Begriffen wurden die Daten vorgestellt, die sich aus den Warnmeldungen von etwa acht Monaten dieses Jahres zusammensetzen. Die Filtermöglichkeiten für diese Warnmeldungen, die bereits in [15] vorgeschlagen wurden, wurden in dieser Arbeit verfeinert und neue Filteroptionen konnten herausgearbeitet werden. Im Anschluss wurde analysiert, welche Daten nach der Anwendung besagter Filtermöglichkeiten noch übrig blieben. Als Resultat gab es vier verschiedene Meldungstypen, auf die ein Fokus gesetzt werden konnte.

Hier wurde zunächst der bestehende, generische Incident-Handling-Prozess der HAW Hamburg vorgestellt. Im Anschluss wurden die existierenden Prozessdiagramme dieses Prozesses um vier weitere Diagramme erweitert, die jeweils die Playbooks der übrigen Meldungstypen visualisieren. Zudem wurde die erste Phase des generischen Incident-Handling-Prozesses angepasst, um die Filtermöglichkeiten für die AW-Meldungen mit

aufzunehmen. Diese Playbooks können nun als Grundlage für eine weitere Automatisierung der zugrundeliegenden Prozesse dienen. Im Anschluss wurde noch ein Blick auf die Implikationen für die Praxis geworfen. Es wurde besprochen, welche gravierenden Auswirkungen bereits die Implementierung der Filtermöglichkeiten darstellt und was nächste Schritte für die Automatisierung der Prozesse sein können.

Diese gesamte Arbeit stellt somit die Basis für eine solche Automatisierungsstrategie dar. In einer einzelnen Arbeit ist es nicht möglich, die Notwendigkeit für die Automatisierung der HAW Hamburg und die generelle Automatisierung von Incident-Response-Prozessen zu lösen. Dies war auch nicht das Ziel der Arbeit. Stattdessen ging es darum, beispielhaft zu zeigen, wie die ersten Schritte für eine solche Automatisierung aussehen können und welche umfassenden Implikationen sich bereits aus diesen ersten Schritten ergeben.

6.2 Limitationen

Diese Arbeit weist einige Limitationen auf, die im folgenden Abschnitt besprochen werden. Hierzu zählen beispielsweise einige Irritationen, die womöglich beim Lesen dieser Arbeit auftreten. Ein Beispiel hierfür zeigt sich in den Playbooks.

Die spezifischen Playbooks, die in dieser Arbeit als Prozessdiagramme dargestellt wurden, brechen mit der bisherigen Struktur. Der generische Incident-Handling-Prozess, der bereits vor dieser Arbeit bestand, gliedert sich sehr klar in vier verschiedene Phasen. Diese Gliederung lässt sich nicht nur anhand der verschiedenen Aufgaben beobachten. Vielmehr gibt es für jede Phase getrennte Prozessdiagramme, die diese Gliederung stärker beleuchten. Die in dieser Arbeit vorgestellten Prozessdiagramme weichen hiervon ab. Zwar bestehen weiterhin eigene Prozessdiagramme für die Phasen *Detect and Report* und *Learn Lessons*, die Übergänge zwischen *Assess and Decide* und *Respond* wurden allerdings aufgeweicht. Der Grund hierfür ist, dass in dieser Arbeit stärker zwischen den verschiedenen Meldungstypen und den resultierenden Playbooks und weniger zwischen den unterschiedlichen Phasen getrennt werden sollte. Der tatsächliche Prozess ist fließend, insofern wurde sich bei der Erstellung der Prozessdiagramme wenig auf diese Unterscheidung fokussiert. Jeweils ein Prozessdiagramm für ein Playbook schien auf konzeptueller Ebene leichter zu verstehen. Sobald die Playbooks in zukünftigen Arbeiten jedoch verfeinert werden, kann es durchaus sinnvoll sein, um einer steigenden Komplexität entgegenzuwirken, wieder eine klarere Trennung nach Phasen einzuführen, indem unterschiedliche Prozessdiagramme für unterschiedliche Phasen existieren.

Ein weiterer Punkt, der bereits in Abschnitt 4.2.2 aufgegriffen wird, ist die Frage nach dem gewählten Format für die Playbooks. In Abschnitt 2.2.2 wird beschrieben, dass Playbooks meist in gängigen Formaten wie JSON, YAML oder XML auftreten. In dieser Arbeit wurde sich dennoch für BPMN entschieden. Der Grund hierfür ist, dass BPMN primär zu Visualisierung des Prozesses genutzt wird, in der Praxis geht es bei Playbooks vor allem um die Maschinenlesbarkeit. Welche Notation für die grafische Darstellung gewählt wird, steht weniger im Fokus. Da die Playbooks dieser Arbeit jedoch noch nicht für ein tatsächliches Tool verfasst wurden, ergibt es mehr Sinn, sich daran zu orientieren, was bereits vorliegt und was leicht visualisiert verstanden werden kann. Da der generische Incident-Handling-Prozess bereits in BPMN notiert wurde, war es somit naheliegend für die spezifischeren Prozesse, die sich darin einbetten, ebenfalls BPMN als Notation zu wählen.

Ein Punkt, der ebenfalls erläutert werden muss, ist der Titel dieser Arbeit: 'Automatisierung von Incident-Response-Prozessen'. Beim Lesen dieser Arbeit fällt eventuell auf, dass der Begriff *Incident-Response-Prozess* seltener fällt als der Begriff *Incident-Handling-Prozess*. Wie bereits in den Grundlagen erwähnt, umschließt das Incident Handling die Incident Response. Es lässt sich darüber streiten, ob der Fokus dieser Arbeit auf den spezifischen Playbooks liegt, die vornehmlich die Incident Response beinhalten, oder ob die Filtermöglichkeiten, die sich eher dem Incident Handling zuschreiben lassen, das Incident Handling als Ganzes ins Zentrum dieser Arbeit rücken. In den vorangegangenen Arbeiten wurde der Fokus spezifischer auf die Incident Response gelegt. Da diese Arbeit als Teil einer Reihe angesehen werden kann, lehnt sich der Titel *Automatisierung von Incident-Response-Prozessen* somit nicht zufällig an Titel wie *Tooling für Incident Response* an. Mit dem stärkeren Fokus auf den generischen Incident-Handling-Prozess der HAW Hamburg, der auf der Arbeit von Kowoll in [12] fußt, verschob sich auch für diese Arbeit der Schwerpunkt. Ein alternativer Titel wie *Automatisierung von Incident-Handling-Prozessen* wäre demnach ebenfalls denkbar.

6.3 Ausblick

Ein detaillierter Blick auf die Schritte, die an diese Arbeit anschließen müssen, um die Automatisierung des Incident Handling an der HAW Hamburg voranzutreiben, wurde bereits in Abschnitt 5.2 geworfen. Kurz zusammengefasst kann jedoch gesagt werden,

dass es zunächst zwei Dinge braucht: Zum einen muss die technische Infrastruktur geschaffen werden, um eine derartige Automatisierung umzusetzen. Viel wichtiger jedoch ist das Verständnis für die bereits bestehenden Strukturen noch weiter zu verschärfen. Die grundlegenden Playbooks, die die spezifische Bearbeitung der Meldungstypen darstellen, reichen hierfür nicht aus. Es muss in engen Kontakt mit den CSIRTs getreten werden, um zu verstehen, wie die aktuellen Prozesse genau ablaufen und wie die Brücke zur Automatisierung gebaut werden kann. Dazu müssen die Prozesse sukzessive verfeinert werden.

Die in dieser Arbeit erarbeiteten Filteroptionen sollten nicht nur als Möglichkeit für die HAW Hamburg gesehen werden, ihre eigenen Prozesse zu optimieren. Auch andere Systeme, die AW-Meldungen des DFN-CERTs beziehen, können von den vorgestellten Ideen profitieren. Die immense Anzahl an Meldungen, die an der HAW Hamburg nicht relevant für den weiteren Incident-Handling-Prozess sind, sind sicher kein Sonderfall der HAW Hamburg. Entsprechend sollte auch ein weiterer Blick zum DFN-CERT selbst gehen. Einige der hier vorgestellten Filteroptionen könnten nicht nur lokal bei den betroffenen Systemen implementiert werden. Fehlmeldungen nach dem Filteransatz für *Doppelte Meldungen* etwa betreffen auch weitere User von AW-Meldungen und es ergibt Sinn, hierfür eine zentralisierte Lösung zu finden.

An dieser Stelle sollte betont werden, dass es nicht möglich sein wird, den aktuellen Incident-Handling-Prozess der HAW Hamburg vollständig oder auch nur weitgehend zu automatisieren. Wie bei vielen Systemen in großen Organisationen sind die bestehenden Prozesse vermutlich zu organisch gewachsen, wodurch ein großer Komplex entstanden ist, der zu starr ist, um ihn direkt auf eine Automatisierungsstrategie umzumünzen. Hier gibt es starke Parallelen zu einer organisch gewachsenen Codebase, die erst in mühsamer Kleinstarbeit refactored werden muss, um sie flexibel genug für eine wirkliche Veränderung zu gestalten. Das Gleiche gilt für die HAW Hamburg. Es reicht nicht aus, Teilaufgaben zu automatisieren. Um wirklich von den Vorzügen der Automatisierung zu profitieren, müssen alle bestehenden Prozesse und Technologien hinterfragt werden. Eine einfache Migration zur Automatisierung ist nicht möglich. Es braucht eine echte Transformation.

Eine erste solche Transformation gründet sich bereits in dieser Arbeit. Mit der immensen Reduzierung an Warnmeldungen, die abgearbeitet werden müssen, wandelt sich das komplette Incident Handling für diesen Teilbereich, da der Scope schlagartig von Arbeitsstunden *täglich* auf Arbeitsstunden *monatlich* erweitert wird. Nur durch Transformationen wie diese kann sich echte Veränderung tiefgreifend vollziehen.

Literaturverzeichnis

[1] Adamek, Sascha ; Laufer, Daniel: Hamburger Hochschule wird erpresst. In: *Tagesschau.de* (2023). – URL https://www.tagesschau.de/investigativ/rbb/hackernangriff-haw-vice-101.html. – Abgerufen am 24. November 2024

[2] Alberts, Christopher ; Dorofee, Audrey ; Killcrece, Georgia ; Ruefle, Robin ; Zajicek, Mark: Defining Incident Management Processes for CSIRTs: A Work in Progress / Carnegie Mellon University. Oktober 2004. – Forschungsbericht

[3] Applebaum, Andy ; Johnson, Shawn ; Limiero, Michael ; Smith, Michael: Playbook oriented cyber response. In: *2018 National Cyber Summit (NCS)* IEEE (Veranst.), 2018, S. 8–15

[4] Bernsen, Rebecca: *Ticket management with OpenProject.* https://www.openproject.org/blog/openproject-for-ticketing-management/. – Abgerufen am 23. November 2024

[5] Bridges, Robert A. ; Rice, Ashley E. ; Oesch, Sean ; Nichols, Jeffrey A. ; Watson, Cory ; Spakes, Kevin ; Norem, Savannah ; Huettel, Mike ; Jewell, Brian ; Weber, Brian u. a.: Testing SOAR tools in use. In: *Computers & Security* 129 (2023), S. 103201

[6] DFN-CERT Services GmbH: *DFN.Security - Logbasierte Usecases.* 2023. – URL https://www.dfn-cert.de/leistungen/security-operations/. – Abgerufen am 12. Juli 2024

[7] FIRST: *FIRST CSIRT Services Framework.* https://www.first.org/standards/frameworks/csirts/csirt_services_framework_v2.1. – Abgerufen am 12. Juli 2024

[8] FIRST: *Team Types Within the Context of Services Frameworks*. https://www.first.org/standards/frameworks/csirts/team-type. – Abgerufen am 12. Juli 2024

[9] González-Granadillo, Gustavo ; González-Zarzosa, Susana ; Diaz, Rodrigo: Security information and event management (SIEM): analysis, trends, and usage in critical infrastructures. In: *Sensors* 21 (2021), Nr. 14, S. 4759

[10] Hochschule für Angewandte Wissenschaften Hamburg: *Informationen für studentische Beschäftigte an der HAW Hamburg*. https://www.haw-hamburg.de/hochschule/hochschuleinheiten/personalservice/informationen-fuer-studentische-beschaeftigte/. 2024. – Abgerufen am 03. November 2024

[11] Hüffmeyer, Marc ; Haupt, Florian ; Leymann, Frank ; Schreier, Ulf: Authorization-aware HATEOAS. In: *CLOSER*, 2018, S. 78–89

[12] Kowoll, L.: *Hauptprojekt: Incident Handling für die Hochschule für Angewandte Wissenschaften Hamburg*. 2024

[13] Kowoll, L.: *Incident Handling für die Hochschule für Angewandte Wissenschaften Hamburg*. 2024

[14] Maute, P.: *Security Orchestration Automation and Response*. 2023

[15] Maute, P.: *Konzepte für Filterstrategien und Playbooks von automatisierten Warnmeldungen*. 2024

[16] Maute, P.: *Tooling für Incident Response*. 2024

[17] Meland, Per H. ; Bayoumy, Yara Fareed F. ; Sindre, Guttorm: The Ransomware-as-a-Service economy within the darknet. In: *Computers & Security* 92 (2020), S. 101762

[18] Metzger, S. ; Hommel, W. ; Reiser, H.: Integriertes Management von Sicherheitsvorfällen. In: *Sicherheit in vernetzten Systemen, 28. DFN-Konferenz*, 2011

[19] MISP Project: *MISP Features and Functionalities*. https://www.misp-project.org/features/. – Abgerufen am 12. Juli 2024

[20] MISP Project: *Who is behind the MISP project?* https://www.misp-project.org/who/. – Abgerufen am 12. Juli 2024

[21] Open Project GmbH: *OpenProject*. https://www.openproject.org/. – Abgerufen am 23. November 2024

[22] Schlette, Daniel ; Empl, Philip ; Caselli, Marco ; Schreck, Thomas ; Pernul, Günther: Do you play it by the books? A study on incident response playbooks and influencing factors. In: *2024 IEEE Symposium on Security and Privacy (SP)* IEEE Computer Society (Veranst.), 2023, S. 60–60

[23] Shadowserver Foundation: *What We Do*. https://www.shadowserver.org/what-we-do/. – Abgerufen am 31. Oktober 2024

[24] Shetty, S. ; Laske, C.: Hype Cycle for ITSM, 2024 / Gartner Research. Juni 2024. – Forschungsbericht

[25] Skopik, Florian ; Settanni, Giuseppe ; Fiedler, Roman: A problem shared is a problem halved: A survey on the dimensions of collective cyber defense through security information sharing. In: *Computers & Security* 60 (2016), S. 154–176

[26] Williams, Amrit ; Nicolett, Mark: Improve it security with vulnerability management. In: *Gartner ID* (2005), Nr. G00127481

A Anhang

A.1 Prozessdiagramme

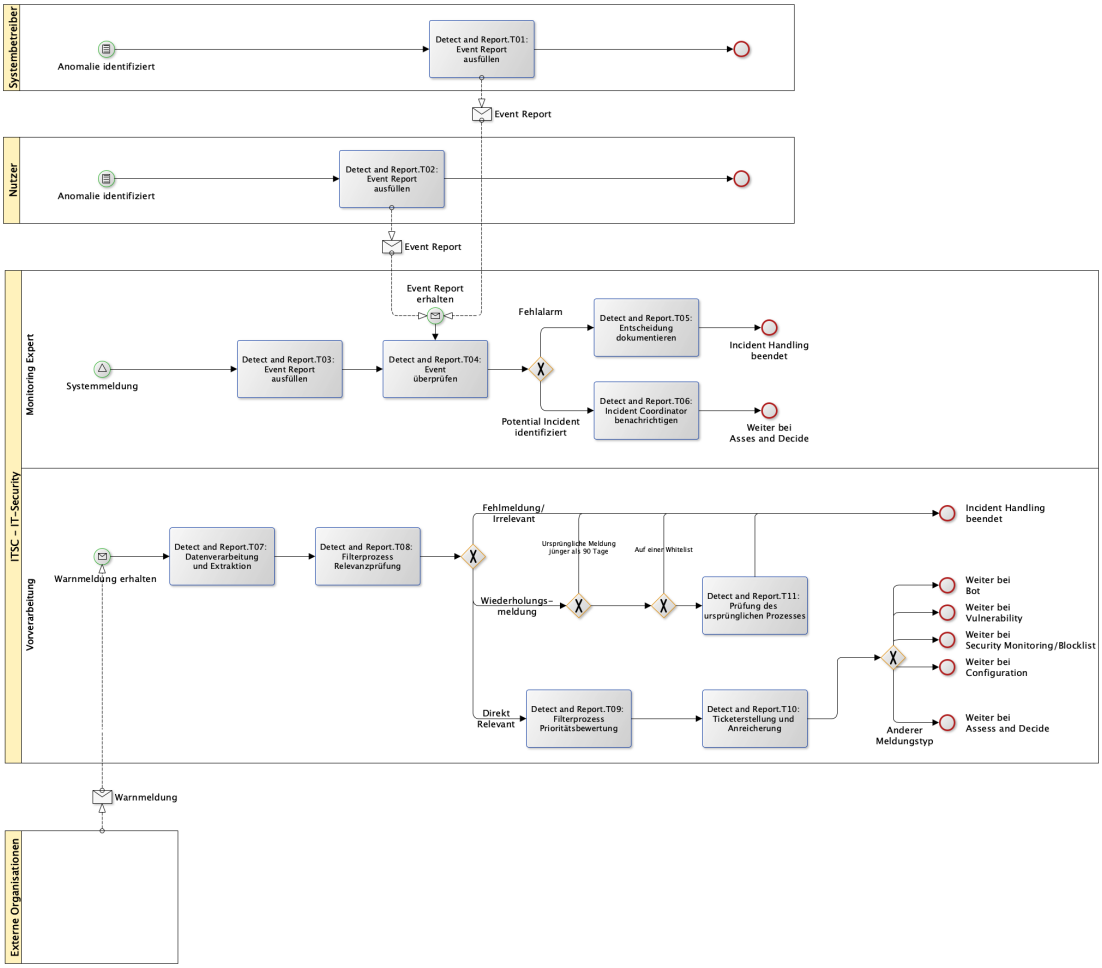

Abbildung A.1: Prozessdiagramm: Detect and Report

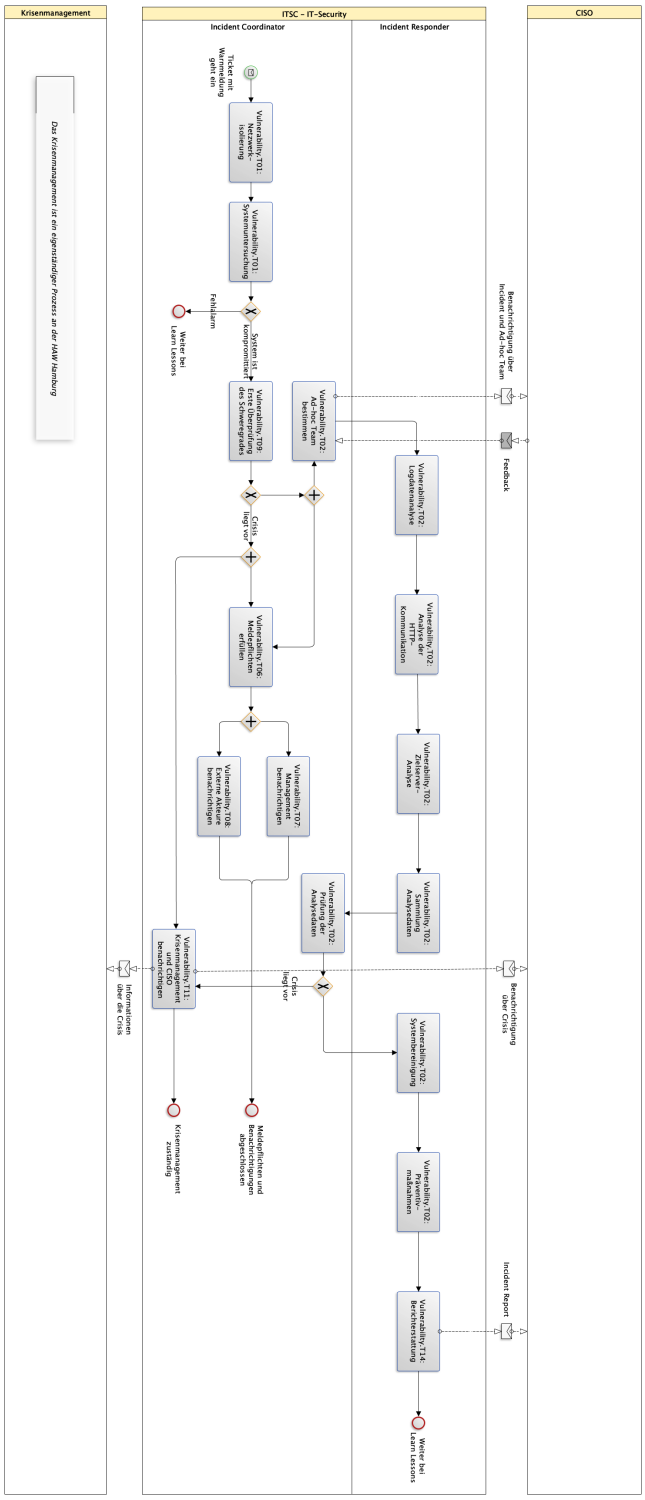

Abbildung A.2: Prozessdiagramm: Bot

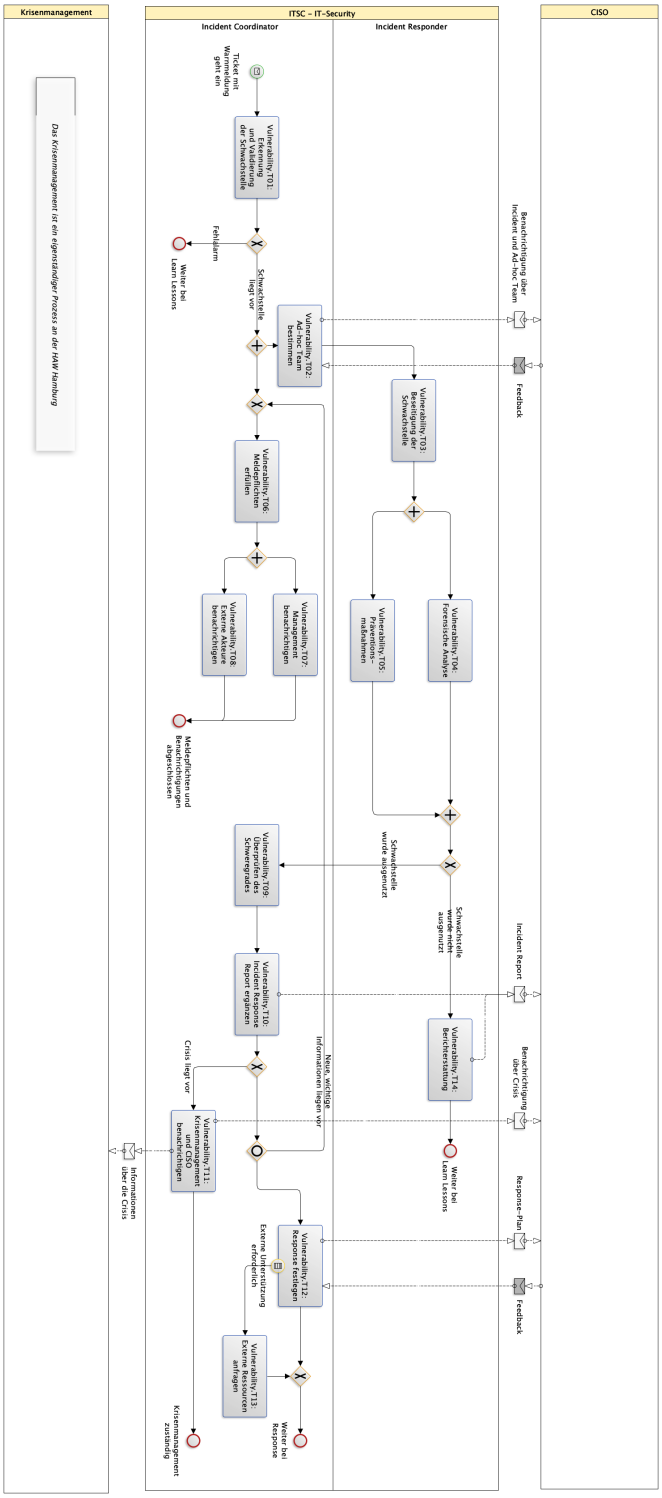

Abbildung A.3: Prozessdiagramm: Vulnerability

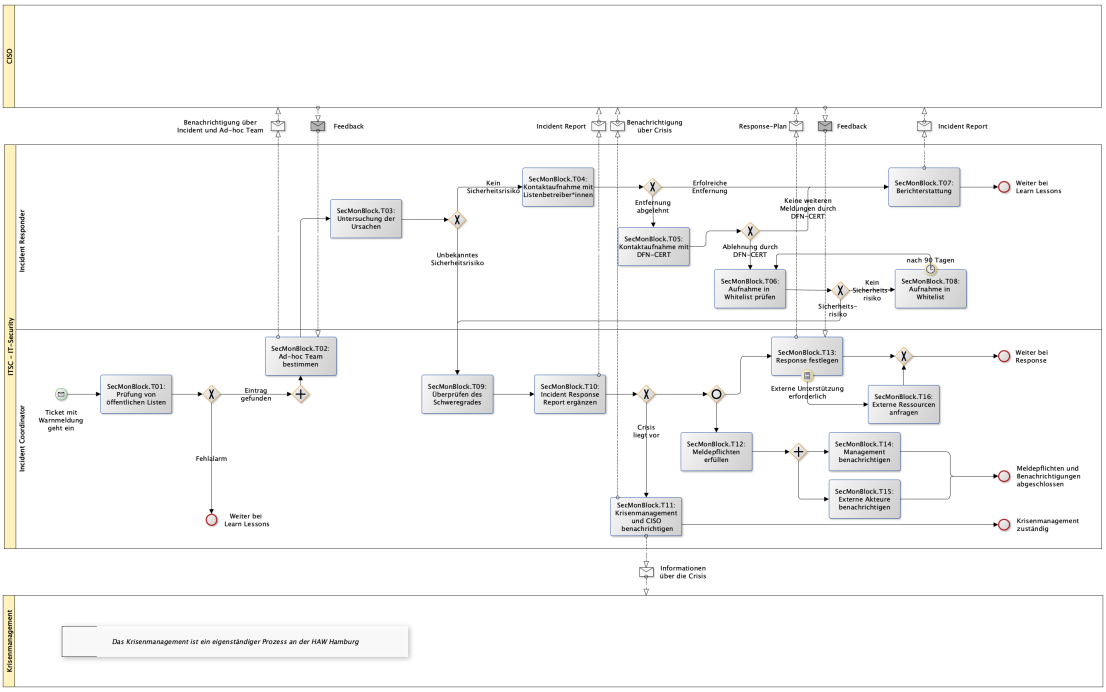

Abbildung A.4: Prozessdiagramm: Security Monitoring/Blocklist

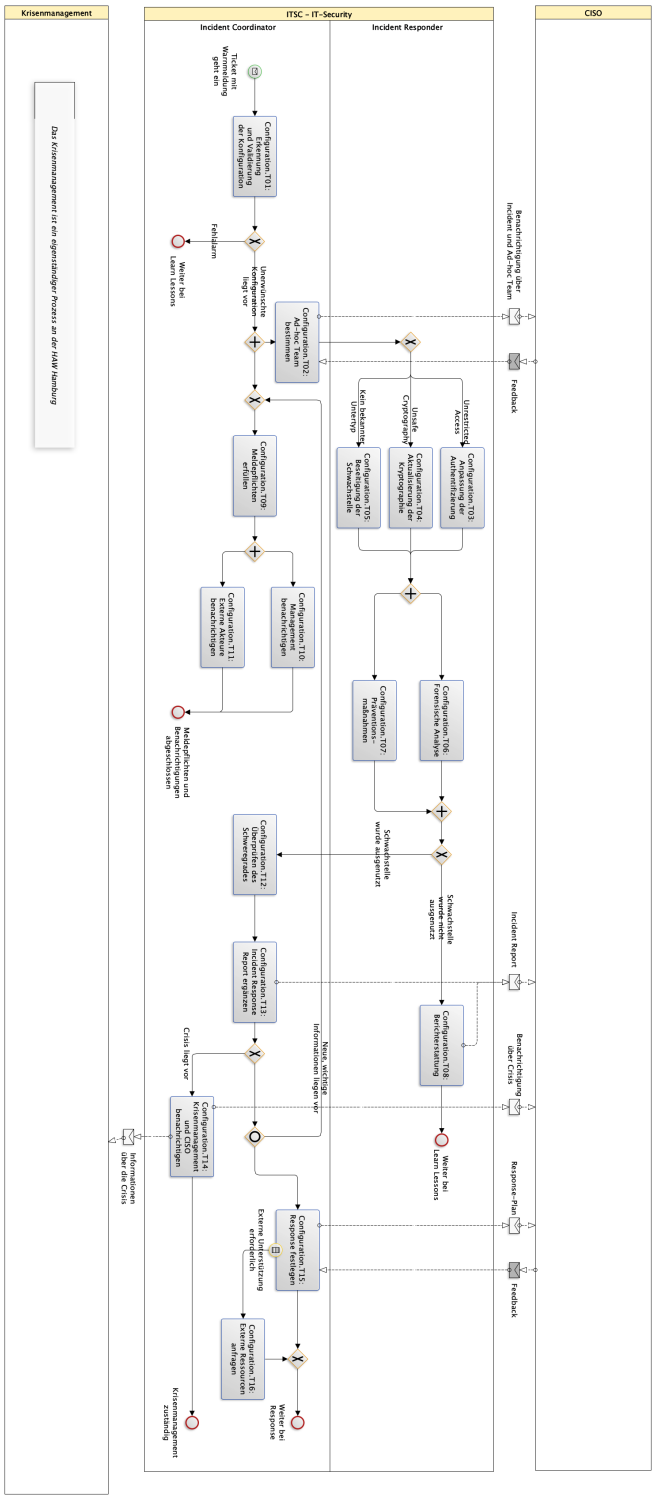

Abbildung A.5: Prozessdiagramm: Configuration

A.2 XML-Schema-Datei für Warnmeldungen

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
   elementFormDefault="qualified" targetNamespace="https://portal.
   cert.dfn.de/2014/aw/" xmlns:aw="https://portal.cert.dfn.de/2014/aw
   /">
  <xs:element name="warning" type="aw:warningType"/>
  <xs:complexType name="warningType">
    <xs:sequence>
      <xs:element minOccurs="0" maxOccurs="unbounded" ref="aw:message
         "/>
    </xs:sequence>
    <xs:attribute name="version" use="required">
      <xs:simpleType>
        <xs:restriction base="xs:token">
          <xs:enumeration value="4.0"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:attribute>
    <xs:attribute name="id" use="required" type="aw:idType"/>
  </xs:complexType>
  <xs:element name="message" type="aw:messageType"/>
  <xs:complexType name="messageType">
    <xs:choice minOccurs="0" maxOccurs="unbounded">
      <xs:element ref="aw:event"/>
      <xs:element ref="aw:description"/>
    </xs:choice>
    <xs:attribute name="category" use="required" type="xs:string"/>
    <xs:attribute name="ip" type="aw:IPType"/>
    <xs:attribute name="domain" type="xs:string"/>
  </xs:complexType>
  <xs:element name="event" type="aw:eventType"/>
  <xs:element name="description" type="xs:string"/>
  <xs:group name="infoType">
    <xs:sequence>
      <xs:element minOccurs="0" ref="aw:record"/>
    </xs:sequence>
  </xs:group>
  <xs:element name="record" type="xs:string"/>
```

```
35  <xs:attributeGroup name="infoType">
36    <xs:attribute name="observation_end" type="xs:dateTime"/>
37    <xs:attribute name="source_port" type="aw:portNumberType"/>
38    <xs:attribute name="source_url" type="xs:string"/>
39    <xs:attribute name="target_ip" type="aw:IPType"/>
40    <xs:attribute name="target_port" type="aw:portNumberType"/>
41    <xs:attribute name="target_url" type="xs:string"/>
42    <xs:attribute name="service" type="xs:string"/>
43    <xs:attribute name="ip_protocol" type="xs:string"/>
44    <xs:attribute name="unknown_url" type="xs:string"/>
45    <xs:attribute name="request" type="xs:string"/>
46    <xs:attribute name="cert_diagnosis" type="xs:string"/>
47    <xs:attribute name="malware_hash" type="xs:string"/>
48    <xs:attribute name="malware_hash_type" type="xs:string"/>
49    <xs:attribute name="occurences" type="xs:string"/>
50    <xs:attribute name="feedback" type="xs:string"/>
51    <xs:attribute name="reporter" type="xs:string"/>
52    <xs:attribute name="certificate_expires_at" type="xs:dateTime"/>
53    <xs:attribute name="source_internal_ip" type="aw:IPType"/>
54  </xs:attributeGroup>
55  <xs:complexType name="eventType">
56    <xs:choice minOccurs="0" maxOccurs="unbounded">
57      <xs:group ref="aw:infoType"/>
58      <xs:element ref="aw:auxiliary_info"/>
59    </xs:choice>
60    <xs:attributeGroup ref="aw:infoType"/>
61    <xs:attribute name="timestamp" use="required" type="xs:dateTime"/
        >
62  </xs:complexType>
63  <xs:element name="auxiliary_info">
64    <xs:complexType>
65      <xs:group ref="aw:infoType"/>
66      <xs:attributeGroup ref="aw:infoType"/>
67    </xs:complexType>
68  </xs:element>
69  <xs:simpleType name="idType">
70    <xs:restriction base="xs:string">
71      <xs:pattern value="\d{4}-\d{3}-\d{4}"/>
72    </xs:restriction>
73  </xs:simpleType>
```

```
74  <xs:simpleType name="IPType">
75    <xs:union>
76      <xs:simpleType>
77        <xs:restriction base="xs:token">
78          <xs:pattern value="
                ((25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-9])\.)
                {3}(25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9][0-9]|[0-9])"/>
79        </xs:restriction>
80      </xs:simpleType>
81      <xs:simpleType>
82        <xs:restriction>
83          <xs:simpleType>
84            <xs:restriction base="xs:token">
85              <xs:pattern value="([0-9a-fA-F]{0,4}:){0,7}[0-9a-fA-F
                    ]{0,4}"/>
86            </xs:restriction>
87          </xs:simpleType>
88          <xs:pattern value="(([0-9a-fA-F]+:){7}[0-9a-fA-F]+)|(([0-9a
                -fA-F]+:)*[0-9a-fA-F]+)?::(([0-9a-fA-F]+:)*[0-9a-fA-F]+)
                ?"/>
89        </xs:restriction>
90      </xs:simpleType>
91      <xs:simpleType>
92        <xs:restriction>
93          <xs:simpleType>
94            <xs:restriction base="xs:token">
95              <xs:pattern value="([0-9a-fA-F]{0,4}:)
                    {0,6}((25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9]).)
                    {3}(25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9])"/>
96            </xs:restriction>
97          </xs:simpleType>
98          <xs:pattern value="(([0-9a-fA-F]+:){6}|(([0-9a-fA-F]+:)
                *[0-9a-fA-F]+)?::([0-9a-fA-F]+:)*)
                ((25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9]).)
                {3}(25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9])"/>
99        </xs:restriction>
100     </xs:simpleType>
101   </xs:union>
102 </xs:simpleType>
103 <xs:simpleType name="portNumberType">
```

```
104     <xs:restriction base="xs:integer">
105       <xs:minInclusive value="0"/>
106       <xs:maxInclusive value="65535"/>
107     </xs:restriction>
108   </xs:simpleType>
109 </xs:schema>
```

Listing A.1: XML-Schema für Warnmeldungen

A.3 Verwendete Hilfsmittel

In der Tabelle A.1 sind die im Rahmen der Bearbeitung des Themas der Masterarbeit verwendeten Werkzeuge und Hilfsmittel aufgelistet.

Tabelle A.1: Verwendete Hilfsmittel und Werkzeuge

Tool	Verwendung
LaTeX	Textsatz- und Layout-Werkzeug verwendet zur Erstellung dieses Dokuments
ChatGPT	Die Texte wurden alle persönlich verfasst, aber es wurde ChatGPT verwendet, um beispielsweise die Filteroptionen auszuarbeiten und um generell an diversen weiteren Stellen auf konzeptueller Ebene eine weitere Sichtweise zu gewinnen.
Microsoft PowerPoint	Zur Erstellung von Abbildungen
SankeyMATIC	Zur Erstellung von Sankey-Diagrammen
yEd	Zur Erstellung von Prozessdiagrammen
draw.io	Zur Erstellung von Abbildungen
Numbers	Zur Bearbeitung der Datensätze und Erstellung von resultierenden Diagrammen
Visual Studio Code	Zur Erstellung von Skripten für die Bearbeitung der Daten

Erklärung zur selbständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Hamburg	02.12.2024	
Ort	Datum	Unterschrift im Original