Gottfried Wilhelm
Leibniz Universität Hannover
Fakultät für Elektrotechnik und Informatik
Institut für Praktische Informatik
Fachgebiet Software Engineering

Interviewstudie zum Tracing von sicherheitsrelevanten Artefakten in der Praxis

Interview Study on Tracing of Security-Relevant Artifacts in Practice

Masterarbeit

im Studiengang Informatik

von

Abdurrahman Sekerci

Prüfer: Prof. Dr. rer. nat. Kurt Schneider
Zweitprüfer: Dr. rer. nat. Jil Ann-Christin Klünder
Betreuer: Marc Herrmann

Hannover, 05.06.2024

Erklärung der Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbständig und ohne fremde Hilfe verfasst und keine anderen als die in der Arbeit angegebenen Quellen und Hilfsmittel verwendet habe. Die Arbeit hat in gleicher oder ähnlicher Form noch keinem anderen Prüfungsamt vorgelegen.

Hannover, den 05.06.2024

Abdurrahman Sekerci

Zusammenfassung

Tracing sicherheitsrelevanter Artefakte ist ein wichtiger Aspekt im Softwareentwicklungsprozess, um die Einhaltung von Sicherheitsstandards und die Reduzierung von Sicherheitsrisiken zu gewährleisten. Trotz der Bedeutung von Traceability besteht eine Ungewissheit in der aktuellen Forschung bezüglich der praktischen Umsetzung des Tracings und der Definition sicherheitsrelevanter Artefakte. Diese Masterarbeit zielt darauf ab, die Definition, Methoden, Prozesse und Herausforderungen des Tracings von sicherheitsrelevanten Artefakten in der Praxis zu untersuchen und Empfehlungen zur Verbesserung der Traceability zu geben. Eine qualitative Interviewstudie wurde mit acht Teilnehmern aus verschiedenen Unternehmen durchgeführt, um tiefere Einblicke in die Praxis des Tracings sicherheitsrelevanter Artefakte zu gewinnen. Die Interviews wurden semistrukturiert geführt und die Ergebnisse anschließend systematisch analysiert. Die Studie ergab, dass die Definition sicherheitsrelevanter Artefakte stark von der jeweiligen Domäne und den spezifischen Phasen des Entwicklungsprozesses abhängt. Sicherheitsstandards wie ISO 27001 sind von zentraler Bedeutung, und der Shift-Left-Ansatz wird zunehmend integriert. Es wurde festgestellt, dass etwa die Hälfte der Unternehmen keine Unterscheidung zwischen sicherheitsrelevanten und nicht-sicherheitsrelevanten Artefakten macht, während die andere Hälfte diese Differenzierung vornimmt. Tools wie Jira, Polarion und JAMA werden häufig zur Unterstützung des Tracings verwendet. Compliance und Audits tragen zur Einhaltung von Sicherheitsstandards bei, stellen jedoch Herausforderungen hinsichtlich der Verwaltung und Sicherheit der Tracing-Daten dar. Die Automatisierung des Tracings bleibt jedoch begrenzt und anfällig für Komplexitäts- und Fehlerprobleme. Die Arbeit empfiehlt z.B. die Schaffung von Bewusstsein für Sicherheitsstandards, die Implementierung automatisierter Tracing-Methoden und die Förderung interdisziplinärer Zusammenarbeit, um die Traceability sicherheitsrelevanter Artefakte zu verbessern.

Abstract

Interview Study on Tracing of Security-Relevant Artifacts in Practice

Tracing security-relevant artifacts is an important aspect of the software development process to ensure compliance with security standards and reduce security risks. Despite the importance of traceability, there is uncertainty in current research regarding the practical implementation of tracing and the definition of safety-relevant artifacts. This master's thesis aims to investigate the definition, methods, processes, and challenges of tracing security-relevant artifacts in practice and to provide recommendations for improving traceability. A qualitative interview study was conducted with eight participants from various companies to gain deeper insights into the practice of tracing security-relevant artifacts. The interviews were semi-structured, and the results were systematically analyzed. The study found that the definition of security-relevant artifacts heavily depends on the specific domain and phases of the development process. Security standards such as ISO 27001 are of central importance, and the shift-left approach is increasingly being integrated. It was observed that about half of the companies do not distinguish between security-relevant and non-security-relevant artifacts, while the other half do make this distinction. Tools like Jira, Polarion, and JAMA are frequently used to support tracing. Compliance and audits contribute to adhering to security standards but pose challenges regarding the management and security of tracing data. However, the automation of tracing remains limited and prone to complexity and error issues. The thesis recommends raising awareness of security standards, implementing automated tracing methods, and promoting interdisciplinary collaboration to improve the traceability of security-relevant artifacts.

Inhaltsverzeichnis

Kapitel 1

Einleitung

Traceability (dt. Verfolgbarkeit) spielt eine wichtige Rolle in der Softwareentwicklung. Sie ermöglicht allen am Softwareentwicklungsprozess Beteiligten, wie z.B. Entwicklern, Testern, aber auch Stakeholdern, die Zusammenhänge zwischen den einzelnen Artefakten besser nachzuvollziehen [82]. Dank der erhöhten Nachvollziehbarkeit können Entwickler, Tester etc. Änderungen oder das Hinzufügen neuer Artefakte unkomplizierter umsetzen. Zusätzlich führt es zu weniger Fehlern und geringerem Arbeitsaufwand, der nötig ist, um die Fehler im Nachhinein zu korrigieren [55]. Trotz dieses großen Stellenwerts, den das Tracing dieser Artefakte hat, verläuft die praktische Umsetzung nicht ohne Probleme.

1.1 Motivation

Das Tracing von sicherheitsrelevanten Artefakten ist ein kritischer Aspekt in der Sicherheit von IT-Systemen. Anforderungen, die Sicherheitseigenschaften wie Verfügbarkeit und Integrität gewährleisten, müssen z.B. rückverfolgbar sein auf Implementierungen, die Sicherheitskonzepte umsetzen. Die Rückverfolgbarkeit von Artefakten, die sicherheitsrelevant sind, sind umso bedeutender als bei generischen Artefakten, da Authentifizierungsmechanismen, Verschlüsselungstechniken, Zugriffskontrollsysteme etc. sich im ständigen Wandel befinden und da Bedrohungsszenarien sich über die fortlaufende Zeit dynamisch ändern [14]. Die fehlende oder fehlerhafte Umsetzung von Traceability ist deswegen direkt mit Sicherheitsrisiken verbunden. Ein effektives Tracing ermöglicht es Organisationen, diese Risiken zu identifizieren, zu überwachen und entsprechende Maßnahmen zu ergreifen, um potenzielle Sicherheitsverletzungen zu verhindern [94]. Folgend daraus kann die Vernachlässigung von Tracing im schlimmsten Falle zu finanziellen oder personellen Schäden führen und auch das Vertrauen zu den Stakeholdern kann negativ beeinträchtigt werden. In sicherheitskritischen Domänen, wie der Gesundheitspflege und Automobilindustrie wird Traceability auch in

der formalen Ausführung versucht umzusetzen. In allen Teilschritten eines Softwareentwicklungsprozesses müssen Anforderungen, die z.B. durch strenge Compliance-Richtlinien wie ISO 27001 [91], eine internationale Norm für Informationssicherheits-Managementsysteme, oder HIPAA (Health Insurance Portability and Accountability Act), ein US-amerikanisches Gesetz, welches den Schutz von Gesundheitsdaten regelt, [73], definiert sind, strikt eingehalten werden. Die ISO 27001 legt z.B. fest, dass alle Anforderungen rückverfolgbar sind, dass Traceability-Informationen konsistent und vollständig sind, dass effektive Prozesse zur Verwaltung eingeführt werden und dass alle Schritte im Softwareentwicklungsprozess dokumentiert und Traceability-Informationen verifiziert werden [91]. Auch wenn nun theoretische Überlegungen wie in der Form von Compliance Richtlinien gemacht wurden, ist nicht davon auszugehen, dass die Umsetzung dieser Richtlinien ohne Probleme verläuft. Die manuelle Erstellung und Instandhaltung von Trace Links ist ein zeitaufwendiger und auch fehleranfälliger Prozess [39]. Dies führt dazu, dass einige Softwareprojekte unvollständige und auch fehlerbehaftete Trace Links haben [56]. Ein weiteres Problem ist das Tooling. Für Tracing werden verschiedenste Tools und Software genutzt. Ein Beispiel ist DOORS (Dynamic Object-Oriented Requirements System), welches dafür genutzt wird, um Anforderungen zu erfassen, verwalten und unter anderem auch nachzuverfolgen [42]. Nun gilt auch, dass in vielen Projekten für verschiedene Artefakte, inklusive des Tracings, verschiedene Tools genutzt werden [88]. Dies hat zur Folge, dass es aufgrund der unterschiedlichen Eigenschaften der verschiedenen Werkzeuge schwierig ist, Tracing-Links zu erstellen, die einem einheitlichen Format folgen [24]. Fehlende Verständnis für die Anwendung von Traceability seitens der Entwickler, Tester etc. ist ein weiteres Problem [6]. Während diese Probleme und allgemeine Aspekte der Traceability in der Forschung gut untersucht sind, fehlt ein Überblick über die Traceability sicherheitsrelevanter Artefakte in der Industrie.

1.2 Lösungsansatz

Um diese notwendigen Erkenntnisse aus der Praxis zu gewinnen, wird eine Interviewstudie mit verschiedenen Akteuren (Entwicklern, Managern, Tester etc.) in der Softwareentwicklung aus verschiedenen Unternehmen (inklusive Unternehmen in sicherheitskritischen Domänen) durchgeführt. Die Interviewstudie bietet sich besonders gut zum Erkenntnisgewinn an, da sie einen tiefergehenden Einblick von Personen ermöglicht, die Tracing nicht nur theoretisch in Kenntnis genommen haben, sondern auch aktiv angewendet haben und eine gewisse Menge an Erfahrung besitzen. Hierbei besitzen die Interviewten im Idealfall auch ein tiefgreifendes Verständnis von diversen anderen Problembereichen in der Softwareentwicklung. Dies ermöglicht uns Einblick in Aspekte bezüglich der Problemstellung zu bekommen,

die nicht mittels einer theoretischen Ausarbeitung möglich wären, da ein Raster aufgebaut wird, welches auf das Problem zugeschnitten ist. Zudem bietet das Interview als Format die Möglichkeit, Fragen, mit denen wir Erkenntnisse gewinnen wollen, an den Kenntnisstand des Interviewten anzupassen und dynamisch abzuändern (im semistrukturierten Format). Damit wir die Ergebnisse der Interviews, zielorientiert eingruppieren und analysieren können, werden nachfolgend zwei Forschungsfragen aufgestellt, die wir in dieser Abschlussarbeit beantworten. Die erste Forschungsfrage lautet:

Forschungsfrage 1:
Was sind sicherheitsrelevante Artefakte in der Praxis?

Der Begriff der „Sicherheitsrelevanz“ hat keine einheitliche Definition, da die Akteure in der Welt der Softwareentwicklung die Relevanz selbst festlegen können. In der Theorie gilt, dass jedes Artefakt, welches einen Einfluss auf die Vertraulichkeit, Integrität und Verfügbarkeit des Gesamtsystems der Software hat, eine Sicherheitsrelevanz besitzt [90]. Zudem gibt es den Aspekt der Safety, auch genannt funktionale Sicherheit. Diese bezieht sich auf den Zustand eines Systems, in dem es frei von unvertretbaren Risiken ist, die durch Fehlfunktionen oder Ausfälle verursacht werden könnten [53]. Es ist in der Praxis aber nicht konkret festgelegt, wann z.B. ein Entwickler oder ein ganzes Unternehmen ein Artefakt als sicherheitsrelevant einstuft und wann dies nicht getan wird, da es unterschiedliche Richtlinien und Prozesse innerhalb der Unternehmen gibt. Unternehmen, z.B. in der Automobilindustrie, folgen strikteren Richtlinien für Softwaresysteme, was zu einer größeren Menge an sicherheitsrelevanten Artefakten und einer strengeren Einstufung führt. Daher ist es wichtig, Personen aus verschiedenen Unternehmen zu befragen, was als sicherheitsrelevant gilt und welche Faktoren für die Einstufung entscheidend sind. Die Eingrenzung innerhalb der bestehenden Artefakte hilft uns, eine Grundlage für die zweite Forschungsfrage zu schaffen und führt die erste Forschungsfrage weiter aus:

Forschungsfrage 2:
Wird Tracing für diese Artefakte betrieben und wenn ja, wie?

Da diese Frage vielschichtig ist und mehrere Unteraspekte thematisiert, teilen wir sie nochmal in 3 Unterfragen auf:

Forschungsfrage 2.1:
Wird Tracing für diese Artefakte betrieben?

Traceability an sich ist eine Eigenschaft, die von Entwicklern oft vernachlässigt wird. Studien zeigen, dass Tracing erst in späteren Phasen der Entwicklung durchgeführt wird, da Tracing als Mehraufwand empfunden

wird und die direkten positiven Effekte nicht gesehen werden [28]. Dies ist auf die Probleme in Abschnitt 1 zurückzuführen. Es wäre daher hilfreich zu wissen, in welchem Umfang Tracing durchgeführt wird und in welchem nicht. Aber nicht nur die Häufigkeit, mit der Tracing durchgeführt wird, ist wichtig, sondern auch die Art und Weise, wie Tracing durchgeführt wird:

Forschungsfrage 2.2:
Mit welchen Methoden und Prozessen wird dieses Tracing durchgeführt?

Die genaue Durchführung des Tracings ist ein zentraler Aspekt dieser Forschungsfrage, um die Wirksamkeit und praktische Anwendung der Traceability zu erfassen. Dabei werden folgende Aspekte untersucht:

- **Verwendung von Traceability-Tools oder Softwarelösungen:** Diese Software erleichtert die Verwaltung von Artefakten und verbessert die Nachverfolgbarkeit [86]. Es wird untersucht, welche Software eingesetzt wird und in welchem Kontext.
- **Compliance und Audits:** Hier geht es um die Einhaltung von Sicherheitsstandards und Compliance-Anforderungen im Tracing, insbesondere in regulierten Branchen wie Finanzwesen, Gesundheit und Verteidigung, wie schon in 1.1 beschrieben. Es wird untersucht, wie Unternehmen das Tracing zur Einhaltung von Vorschriften anwenden.
- **Prozessintegration:** Da die Softwareentwicklung ein mehrstufiger Prozess ist, muss das Tracing an bestehende Teilschritte angepasst werden, um Konflikte zu vermeiden. Untersucht wird, wie Tracing in Entwicklungs- und Qualitätssicherungsprozesse integriert wird und wie es mit Sicherheitskomponenten interagiert.
- **Automatisierung:** Tracing kann teilweise oder vollständig automatisiert werden, was zusätzliche Arbeit und Wartung erfordert. Es wird erfasst, wie und was automatisiert wurde.
- **Zusammenarbeit und Kommunikation:** Da Nutzerinteraktion im Tracing essenziell ist, wird untersucht, welche Schritte oder Methoden die Zusammenarbeit erleichtern und wie Rollenunterschiede aussehen.

Forschungsfrage 2.3:
Welche Probleme bestehen bei diesem Tracing?

Das Tracing in der Praxis ist mit zahlreichen Problemen verbunden, wie auch schon in Abschnitt 1 und 1.1 erläutert. Dazu zählen die Komplexität der Artefakte, welche das Tracing durch Subkomponenten erschwert, die Kosten und die Ressourcen, die für das Tracing benötigt werden, der individuelle Widerstand der Teilhabenden im Softwareentwicklungsprozess

gegen das Tracing, die Datenqualität der Tracing-Daten und insbesondere Datenschutzmissstände und Sicherheitsrisiken, die bei dem Tracing von sicherheitsrelevanten Artefakten auftreten können. Es wäre vorteilhaft, auch potenzielle Probleme im Interview zu erfragen, um im weiteren Verlauf der Arbeit Lösungsansätze zu generieren. Basierend auf den Ergebnissen der Diskussion im Rahmen dieser Forschungsfragen, wollen wir zum Schluss ein Modell entwickeln. Das Modell soll dazu dienen, Praktiken und Empfehlungen zur Verbesserung der Traceability für sicherheitsrelevante Artefakte ableiten zu können. Unser Ziel ist es, durch Erkenntnisse aus der Praxis und verschiedenen Entwicklungsumgebungen einen Leitfaden für die praktische Umsetzung zu erstellen. Diese manifestieren sich in der Form von Handlungsempfehlungen.

1.3 Struktur der Arbeit

Diese Abschlussarbeit ist in mehrere Kapitel gegliedert. In Kapitel 2 werden die grundlegenden Konzepte der Rückverfolgbarkeit und der Sicherheitsrelevanz näher erläutert. Verwandte Arbeiten zu dem Thema des Tracings von sicherheitsrelevanten Artefakten werden in Kapitel 3 vorgestellt. Anschließend wird in Kapitel 4 das Interview auf konzeptioneller Ebene erläutert. Dazu wird die Form des Interviews näher erläutert und der Interviewleitfaden beschrieben. Des Weiteren werden Informationen über die Zielgruppe des anonymisierten Interviews gegeben und wie die Ergebnisse des Interviews ausgewertet werden. Die Ergebnisse der Interviews und nachfolgenden Analyse werden in Kapitel 5 dargestellt. Die Forschungsfragen werden in Kapitel 6 durch die Diskussion und Einordnung der Ergebnisse beantwortet. Aus den Ergebnissen werden dann im selben Kapitel Handlungsempfehlungen abgeleitet. Im letzten Kapitel 7 wird ein abschließendes Fazit gezogen und ein Ausblick auf weitere Forschung im Themenbereich der Abschlussarbeit gegeben.

Kapitel 2

Grundlagen

In diesem Kapitel wird erläutert, was Traceability bedeutet und wie sie umgesetzt wird. Außerdem wird beschrieben, was Sicherheitsrelevanz in der Softwareentwicklung bedeutet.

2.1 Traceability

In der Softwareentwicklung bezeichnet Traceability die systematische Nachverfolgung und Dokumentation von Beziehungen zwischen zwei oder mehr verschiedenen Software-Artefakten während des gesamten Entwicklungszyklus [44]. Somit ist Traceability eine wünschenswerte Eigenschaft für die Softwareentwicklung und führt dazu, dass man die logischen Verbindungen zwischen Artefakten nachvollziehen kann. Zu den Artefakten zählen beispielsweise Anforderungen, Designspezifikationen, die Beschreibung der Softwarearchitektur, Implementierung und Testfälle. Zwei oder mehr Artefakte werden als Trace Link bezeichnet. [18]. Ein Beispiel für einen typischen Trace Link ist die Verbindung zwischen Anforderungen und Code, die in der Literatur auch als 'Requirements-Traceability' bezeichnet wird [36]. Dies ist definiert als die beidseitige Verfolgbarkeit zwischen der Anforderungsphase am Beginn der Softwareentwicklung und zur letzten Phase, dem Deployment der Software [36]. Somit werden Trace Links werden durch die Kette von Artefakten innerhalb des Softwareentwicklungsprozesses aufgebaut. Dabei betont die Zweiseitigkeit, dass nicht nur die Anforderungen zu dem jeweiligen Code zurückverfolgt werden kann, sondern auch von dem Code selbst zur Anforderung [36]. Trace-Links können in verschiedenen Formen dargestellt werden, da als primäre Voraussetzung nur gilt, dass eine semantische Verbindung zwischen mehreren Artefakten hergestellt wird [36]. Eine Darstellungsform ist eine Tabelle oder eine Matrix, die verschiedene Artefakte horizontal und/oder vertikal miteinander verbindet [85]. Auch visuelle Darstellungen wie Graphen oder Diagramme können genutzt werden [9]. Es gibt auch textuelle Darstellungen, bei denen durch

explizite Erwähnung von Artefakten ein Bezug hergestellt wird. Ein Beispiel hierfür sind Kommentare im Code, die auf Anforderungen oder Testfälle verweisen. Diese können auch als Hyperlinks aufgebaut sein, die bei Zugriff auf sie zum referenzierten Artefakt führen.

2.1.1 Tracing-Verfahren

Durch die langjährige Anwendung von Tracing gibt es mittlerweile zahlreiche Verfahren und Werkzeuge, die die Nachverfolgbarkeit von Artefakten erleichtern. Im Folgenden werden diese aufgelistet:

1. **Anforderungsmanagement Software:** Diese Tools sind darauf spezialisiert, Anforderungen und deren Änderungen im Laufe eines Projekts systematisch zu verfolgen. Anforderungen können definiert, dokumentiert und kategorisiert werden. Sie ermöglichen es, Verknüpfungen zwischen Anforderungen und anderen Projektartefakten wie Testfällen, Designspezifikationen und Implementierungscodes herzustellen. Mithilfe von Analysefunktionen ist es möglich, Änderungen innerhalb eines Projekts zu überwachen. Ein Beispiel für eine solche Software ist IBM DOORS [42].

2. **Integration in Entwicklungsumgebungen:** Einige Traceability-Werkzeuge sind in bestehende Entwicklungsumgebungen integriert. Dadurch können Teams in der Softwareentwicklung direkt auf Traceability-Informationen zugreifen, ohne dass separate Tools für das Tracing erforderlich sind. Somit wird das Tracing zu einem natürlichen Teil des Entwicklungsprozesses. Dies erleichtert den Teams den Zugriff auf diese Informationen und fördert die Zusammenarbeit, da in der Regel in derselben Umgebung gearbeitet wird. Zudem ist keine manuelle Eingabe erforderlich, da Informationen aus den Artefakten oder Aktionen von Entwicklern gesammelt werden. Eine Form solcher Integrationen sind Plug-ins, wie zum Beispiel Eclipse Mylyn [50], ein Plug-in für die Eclipse IDE. Mit Mylyn können Entwicklungsdokumente wie beispielsweise Dateien für Code-Aufgaben zugewiesen werden. Dadurch wird die Sicht auf das zu Entwickelnde auf die zu erledigenden Aufgaben eingeschränkt, wodurch ein indirekter Tracelink von Aufgaben auf Source-Code oder Tests aufgebaut wird.

3. **Traceability in der Versionskontrolle**: Eine weitere Methode ist es, Versionsverwaltungssoftware, wie Git [19], für die Generierung von Traceability Informationen zu benutzen. Dieser Ansatz ermöglicht es, Änderungen im Code und ihre Beziehungen zu Anforderungen, Tests, Designdarstellungen und anderen Dokumenten genau zu verfolgen. Durch die Verwendung von Commit-Nachrichten und Branch-

Strategien können Teams die Entwicklungsgeschichte eines Projekts klar abbilden und die Herkunft spezifischer Änderungen oder Entscheidungen nachvollziehen. Die Integration von Git in Projektmanagement Tools wie Jira [54] ermöglicht es dann, diese Tracing-Informationen in den Entwicklungsprozess einzubinden.

4. **Automatische Methoden:** In automatischen Methoden werden Algorithmen genutzt, um Verknüpfungen und Abhängigkeiten zwischen verschiedenen Artefakten autonom zu identifizieren und zu verwalten [70]. Diese Methoden basieren auf natürlicher Sprachverarbeitung [70], Information Retrieval [21], Machine Learning [47] oder auch graphenbasierten Verfahren [95]. Dadurch entfällt die Notwendigkeit, dass Entwickler, Tester und andere Beteiligte Zeit für das manuelle Tracing aufwenden müssen. Außerdem erkennt der Algorithmus potenzielle Trace Links, die für den Anwender sonst schwer erkennbar wären [70]. Ein Beispiel für einen solchen Algorithmus ist die Verwendung von Deep Learning, um Code-Dateien anhand von Fehlerberichten zu finden und somit Trace Links zwischen Code und Fehlerbericht aufzubauen [47].

5. **Visualisierungstools:** Um die Verständlichkeit und Übersichtlichkeit zu verbessern, können Visualisierungstools genutzt werden, um grafische Darstellungen der Beziehungen zwischen Artefakten zu erstellen. Diagramme, Graphen und interaktive Schnittstellen helfen den Anwendern, die Struktur und Verbindungen der Artefakte schnell zu erfassen. Ein Beispiel für eine solche Anwendung ist die „Hierarchical Trace Map“ [9]. In diesem Tool werden Artefakte als einzelne, verschiedenfarbige Knoten dargestellt. Jede Farbe repräsentiert einen bestimmten Artefaktentyp. Die Knoten sind miteinander verbunden, wobei diese Verbindungen als Tracing Links interpretiert werden müssen. Das Besondere an diesem Tool ist, dass es verschiedene Sichtmodi gibt. Beispielsweise gibt es einen Filtermodus, um nur bestimmte Artefaktbeziehungen anzuzeigen.

2.1.2 Probleme in Traceability

In der Softwareentwicklung gibt es zahlreiche Probleme bei der Traceability. Die Nützlichkeit, Anwendbarkeit und Vorteile von Traceability werden von den Akteuren (Entwicklern, Testern etc.) oft nicht verstanden [35]. Zudem gibt es das Problem, dass es schwierig ist manuell Trace Links zu verwalten [61, 4]. Im Softwareentwicklungsprozess ändert sich der Artefaktenbestand kontinuierlich. Funktionalitäten werden ständig hinzugefügt, geändert oder entfernt, wodurch bestimmte Artefakte veraltet oder obsolet werden können. Dies kann temporär zu Problemen führen, da auch bestehende Trace Links kontinuierlich an die Änderungen angepasst werden müssen. [4]. Die

Vernachlässigung dieses Prozesses mindert die Vorteile von Trace Links [4]. Außerdem kann es aufgrund der hohen Komplexität vorkommen, dass Trace-Links im Laufe der Zeit vergessen und nicht aktualisiert werden. Dies führt zu fehlerhaften Informationen innerhalb der Trace-Links und stellt die Validität der Trace-Links für die Akteure, die darauf zugreifen, infrage [35]. Das Verwalten von Trace Links ist somit mit hohen zeitlichen Kosten verbunden. Das vorher erwähnte Unverständnis über Tracing ist hierbei ein verstärkender Faktor, da Tracing dadurch vernachlässigt wird und der Korrekturaufwand im Nachgang sich erhöht [35]. Daher besteht das Problem, dass das Tracing aufgrund des Wachstums der Projekte innerhalb der Organisation noch zeitaufwendiger und komplexer wird und somit nicht gut skaliert. Eine Kostenabschätzung, um zu bestimmen, wie viel Tracing notwendig ist und für welche Artefakte es benötigt wird, wäre vorteilhaft. Es ist jedoch schwierig zu bestimmen, welchen Nutzen ein bestimmter Trace-Link hat, da es keine klaren Heuristiken gibt [35]. Dadurch kann überflüssiges und suboptimal gewähltes Tracing durchgeführt werden. Die hohen Kosten und die Schwierigkeit, die Kosten abzuschätzen, verstärken das Verhalten, Tracing erst in den Endphasen eines Projekts einzusetzen. Das Tracing gestaltet sich jedoch umso schwieriger, da das Projekt bereits eine gewisse Größe erreicht hat und das Verständnis der Akteure über das Tracing nicht kontinuierlich aufgebaut wurde [35]. Einer der Hauptgründe für all diese Probleme ist der Faktor Mensch, da dieser die jeweiligen Tracingdaten erstellt und verwaltet. Aus diesem Grund stellt automatisiertes Tracing eine mögliche Alternative dar. Allerdings haben automatisierte Algorithmen aktuell noch das Problem, dass sie keine hohen Präzisionswerte aufweisen [40]. Es gibt viele generierte Trace-Links, die einen Zusammenhang zwischen Artefakten darstellen, wo im praktischen Einsatz keiner existiert [8]. Dies liegt daran, dass automatisierte Algorithmen nicht in der Lage sind, die Semantik der Artefakte zu verstehen und auch kein Domänenwissen mit den Artefakten verknüpfen können. Dadurch entstehen ungenaue Ergebnisse [37]. Diese können jedoch im Nachgang besser nachvollzogen werden, da die Entscheidungen eines automatisierten Algorithmus nach aktuellem Stand nicht immer gut interpretierbar sind [38].

2.2 Secure Software-Engineering

2.2.1 Was ist Sicherheit in Software?

Sicherheit in der Softwareentwicklung bedeutet, dass Software so konzipiert wird, dass sie gegenüber potenziellen Angreifern und Risiken, wie veralteten Technologien oder Schadsoftware, widerstandsfähig ist. Drei wichtige Eigenschaften sicherer Software sind:

- **Vertraulichkeit (Confidentiality)**: Schutz von Daten vor unberech-

tigtem Zugriff und Offenlegung. Dies wird durch Maßnahmen wie Datenverschlüsselung, Zwei-Faktor-Authentifizierung und rollenbasierte Zugriffskontrolle (RBAC) erreicht [90].

- **Integrität (Integrity)**: Schutz von Daten vor unerlaubten Änderungen, um Zuverlässigkeit und Korrektheit der Software sicherzustellen. Beispiele sind die Verwendung von Hash-Signaturen und Versionierungssystemen [90].
- **Verfügbarkeit (Availability)**: Sicherstellung, dass IT-Systeme, Anwendungen und Daten zeitgerecht für Benutzer zugänglich sind. Maßnahmen umfassen Echtzeit-Tracing und Lastenverteilung auf mehrere Serverknoten [90].

Diese Eigenschaften müssen während des Betriebs der Software erfüllt werden, um eine hohe Sicherheit zu gewährleisten. Sicherheit ist aber nicht nur begrenzt auf den Schutz vor Angreifern. Im Englischen ist dies ersichtlich an den Begriffen „Security“ und „Safety“. Im Gegensatz zu Security steht der Begriff Safety, im Deutschen genannt funktionale Sicherheit, für den Schutz vor zufälligen oder unbeabsichtigten Fehlern [53]. Bedrohungen wären hier z.B. physikalischer Natur, wie extreme Umweltbedingungen oder äußere Gewalteinwirkung auf das physikalische Rechenmedium. Aber auch die Vermeidung von Personen- oder Sachschäden durch zufällige Hardwareausfälle gehören dazu [53]. Bei Safety geht es also darum, das System vor unbeabsichtigten Schäden zu schützen, während bei Security die klassischen Sicherheitsattribute wie Vertraulichkeit, Authentizität oder Integrität eine Rolle spielen [53].

2.2.2 SDLC

Secure Software-Engineering (SSE) ist ein fundamentaler Aspekt der zeitgenössischen Softwareentwicklung, der sich mit der Integration von Sicherheitsmaßnahmen in den Softwareentwicklungsprozess befasst. Dabei geht es nicht nur darum, Funktionen zu implementieren, die Sicherheitsfunktionen umsetzen, sondern auch um die systematische Berücksichtigung von Sicherheit als nicht-funktionale Eigenschaft in jeder Phase des Softwareentwicklungslebenszyklus. In der englischen Literatur wird dies auch als Secure Development Lifecycle bezeichnet [77]. In vielen Organisationen wird Sicherheit jedoch nicht als Eigenschaft betrachtet, die Artefakte im SDLC als Attribut besitzen müssen, sondern als Ergänzung, die erst in der Post-SDLC-Phase in Betracht gezogen wird [67]. Zusätzlich gilt, dass Sicherheit mit hohen zeitlichen und finanziellen Kosten assoziiert wird, da es diverse Deadlines und zeitliche Einschränkungen gibt [65], fehlendes Wissen, wie man Sicherheit in den Entwicklungsprozess einbindet [87], und die mangelhafte Setzung von Anforderungen, die Sicherheit thematisieren, womit Sicherheitsaspekte in

den späteren Phasen der Softwareentwicklung nur spärlich umgesetzt werden können [29]. Zudem gibt es das Phänomen, wenn Sicherheit dann doch gewissermaßen beachtet wird im Entwicklungsprozess (und nicht danach), dann wird es nur in der Implementationsphase umgesetzt [51]. Dies kann zu fehlerhaften Ergebnissen führen und Sicherheitsrisiken verursachen [51]. Diese Probleme sind im Grunde alle darauf zurückzuführen, dass Entwickler die Zuständigkeit für die Sicherheitsangelegenheiten nicht bei sich sehen, und somit ignorant über diese sind [48]. Die Nichtbeachtung und Nichtumsetzung des SDLC hat direkte Konsequenzen für die Organisation. Es erhöht die Anfälligkeit für Fehler im Code [71], verursacht hohe Kosten, um im Nachhinein Sicherheitsmängel zu beheben [52], birgt Risiken für sensible Kundeninformationen [75] und es fehlen logischerweise Mechanismen, um sich an neue Sicherheitsrisiken anzupassen.

2.2.3 Ein Beispielprozess

Da wir im vorherigen Abschnitt den SDLC vorgestellt haben, möchten wir in diesem Abschnitt die fünf großen Phasen des Softwareentwicklungsprozesses behandeln, wobei der Fokus auf den Sicherheitsaspekten liegt. Dies hilft dabei, zu verstehen, wie ein SDLC in der Praxis umgesetzt wird.

Anforderungsphase

Die Requirements-Phase im Softwarelebenszyklus bezieht sich auf den Prozess der Ermittlung, Analyse, Dokumentation und Validierung der Bedürfnisse und Anforderungen der Stakeholder für das zu entwickelnde Softwareprodukt [67]. Insbesondere für die Sicherheitseigenschaften wird ein Fundament für die darauffolgenden Phasen gelegt. Am Anfang dieser Phasen werden zunächst Anforderung festgelegt, die die schon vorher in 2.2.1 Eigenschaften (und weitere mögliche) erfüllen. Im Gegensatz zur generischen Requirements-Phase liegt hier ein größerer Fokus darauf, dass Anforderungen auf Basis von Compliance-Richtlinien wie beispielsweise HIPAA und ISO 27001 erstellt werden, da Sicherheitsaspekte und domänenspezifische Erwartungen miteinander verbunden sind. Dies erfordert auch die Einbeziehung von Compliance-Beauftragten und Sicherheitsexperten wie Security-Beratern bei der Erstellung der Anforderungen. Zusätzlich zu den Compliance-Richtlinien müssen auch Anforderungen für die sichere Verarbeitung, Speicherung und Übertragung sensibler Daten erstellt werden, die unter dem Begriff Datenschutz zusammengefasst werden. Die Umsetzung hängt davon ab, in welcher Region die Software verwendet wird. In Deutschland wird der Datenschutz durch die DSGVO geregelt [15]. Abweichend vom generischen Requirements Engineering wird auch die Methode der Risikoanalyse angewendet. Dabei werden potenzielle Risiken identifiziert und kategorisiert. Dies kann mittels qualitativer Methoden

geschehen, die Sicherheitsrisiken anhand von qualitativen Metriken bestimmen (durch das subjektive Empfinden der Requirements Engineers). Ein Beispiel für solch eine Methode ist OCTAVE (Abkürzung für Operationally Critical Threat, Asset, and Vulnerability Evaluation) [3], der Risiken anhand von organisatorischen Bestimmungen und Einschätzungen von Personen innerhalb der Organisation festlegt und priorisiert [10]. Zudem gibt es auch quantitative Methoden, wie z.B. FAIR (Abkürzung von Factor Analysis of Information Risk) [10], der Risiken einen monetären Wert zuschreibt und so Risiken anhand des finanziellen Wertes priorisiert. Abschließend lässt sich festhalten, dass Anforderungen, die Sicherheitsaspekte behandeln, viel dynamischer sind als generische Anforderungen. Sie müssen aufgrund der sich schnell ändernden Bedrohungslandschaft konsequent überprüft und angepasst werden.

Designphase

In der Design-Phase geht es um die detaillierte Ausarbeitung und Planung der Architektur, Komponenten, Schnittstellen und anderer Merkmale der zu entwickelnden Software. Damit wandelt die Design-Phase die optimalerweise vorher erstellten Anforderung in eine Design-Spezifikation um [43]. Eine Architektur in der Design-Phase, die einen Fokus hat auf Sicherheit, ist in die Gesamtarchitektur integriert und erfüllt die in 2.2.1 Eigenschaften durch Sicherheitskomponenten wie Verschlüsselung, Zugriffskontrolle oder auch sicherer Datenübertragung [46]. Nach Goertzel et al. müssen diese Komponenten auch vor Risiken und potenziellen Attacken geschützt werden, indem sie z.B. separat isoliert werden [46]. Um Architekturen mit Sicherheitsfokus besser modellieren zu können, gibt es Modellierungssprachen, die UML als Grundlagen haben, wie z.B. UMLSec [49]. UMLSec integriert Sicherheitsanforderungen direkt in das Design, indem die Notation der UML-Sprache erweitert wird. So können Verschlüsselungsverfahren visuell im Diagramm dargestellt und Einschränkungen bestimmt werden, die Zugriffe auf ein Objekt an eine Bedingung koppeln (Zugriffskontrolle). Zusätzlich zur groben Architektur muss man auch Designprinzipien und Designpatterns beachten. Es gibt hierfür 8 fundamentale Prinzipien [78]:

1. **Least Privilege:** Jeder Prozess sollte nur die minimalen Rechte besitzen, die zur Erfüllung seiner Aufgabe notwendig sind.
2. **Fail-Safe Defaults:** Standardmäßig sollte der Zugriff verweigert werden; Zugriff wird nur bei expliziter Erlaubnis gewährt.
3. **Economy of Mechanism:** Sicherheitsmechanismen sollten einfach und klein sein.
4. **Complete Mediation:** Jeder Zugriff auf Ressourcen sollte stets vollständig kontrolliert werden.

5. **Open Design:** Die Sicherheit eines Systems sollte nicht auf Geheimhaltung basieren, sondern auf der Stärke der Mechanismen.

6. **Separation of Privilege:** Wichtige Aktionen erfordern das Erfüllen mehrerer Bedingungen.

7. **Least Common Mechanism:** Die Anzahl gemeinsam genutzter Mechanismen sollte minimiert werden.

8. **Psychological Acceptability:** Sicherheitsmechanismen sollten benutzerfreundlich gestaltet sein.

Als weiteres Werkzeug im Designprozess gibt es Design-Patterns. Dies sind erprobte Lösungen oder Vorschläge, wie ein Designkonzept gestaltet werden soll oder nicht [59]. Ein mögliches Muster ist beispielsweise die Validierung der Eingabe, um SQL-Injections oder Cross-Site Scripting zu verhindern. Ein weiteres Muster ist das Ausloggen eines Benutzers aus seinem Konto nach einer festgelegten Anzahl von Fehlversuchen bei der Passworteingabe, um Brute-Force-Angriffe zu verhindern [17]. Ein Anti-Pattern, also ein Pattern, welches man in seinem System vermeiden sollte, ist z.B. der Single Point of Failure. Der Single Point of Failure tritt auf, wenn das Versagen einer Komponente im Gesamtsystem zum Versagen des gesamten Systems führt. Letzteres Verfahren, welches sich in der sicherheitsorientierten Design-Phase abhebt, ist das Threat Modeling. Dabei wird ein Modell erstellt, das mögliche Bedrohungen für ein System, eine Anwendung oder einen Prozess analysiert. Daraus werden Gegenmaßnahmen zur Vermeidung oder Minderung der Auswirkungen dieser Bedrohungen vorgeschlagen. Es gibt zahlreiche Modelle für das Threat Modeling, die formal oder grafisch gestaltet sind [93]. Threat-Modelle sollen grundlegend die Fragen beantworten, welches System aufgebaut wird, welche Bedrohungen bestehen und welche Konsequenzen aus den Bedrohungen folgen. Außerdem sollen sie aufzeigen, wie wahrscheinlich die Bedrohungen sind und welche möglichen Lösungsansätze es gibt. Somit können Threat-Modelle helfen, Bedrohungen zu identifizieren und das Design sowie die Anforderungen des Softwaresystems präventiv an die Bedrohungen anzupassen. Ein mögliches Threat Modell sind Attack-Trees [62]. Attack-Trees sind ein diagrammatisches Werkzeug zur systematischen Analyse und Darstellung potenzieller Angriffswege. Der Baum wird hierarchisch aufgebaut, wobei der Wurzelknoten das Hauptangriffsziel darstellt, z.B. der Zugang zu einem Netzwerk. Die Kinderknoten der Wurzel und die folgenden Knoten stellen Methoden oder Strategien dar, die erfüllt sein müssen, um die Bedingungen im Elternknoten zu erfüllen. Zusätzlich können Knoten für die logischen Operationen „OR“ und „AND“ eingebunden werden, um diese auf die Kinderknoten anzuwenden. Mithilfe von Attack-Trees ist es somit gezielt möglich, alle Schritte für einen Angriff sequenziell darzustellen und Gegenmaßnahmen zu entwickeln.

Implementierungsphase

Auch in der Implementierungsphase hat Security in der SSE eine große Relevanz. Um sicherzustellen, dass der Code keine Sicherheitsrisiken birgt, muss er so geschrieben werden, dass er keine Sicherheitslücken aufweist. Es gibt Richtlinien für sicheres Codieren, die genauso wie in der Anforderungsphase einzuhalten sind und Fehler verhindern, die von Entwicklern implementiert werden [79]. Dazu gehört z.B., dass es bei Datenstrukturen zu keinen Bufferoverflows kommen kann, da dies eine gängige Methode ist, um das System, auf dem die Software ausgeführt wird, zu kompromittieren oder sogar fremden Code in andere Anwendungen einzuschleusen [23]. Auch generische Speicherlecks in Sprachen wie C oder C++, die beispielsweise durch unsichere Nutzung von Pointern entstehen, gehören dazu. Es gibt verschiedene Methoden und Prozesse, um Sicherheitsprobleme abzufangen. Im Rahmen von Code-Reviews wird neuer Code von anderen Entwicklern im Team oder in der Organisation selbst untersucht und beurteilt, bis er frei von Risiken ist [12]. Zusätzlich zu den Code Reviews gibt es statische Code Analyse Tools. Diese sind teilweise in den gängigen IDE's mitintegriert, und zeigen bestehende Sicherheitslecks schon früh im Build Prozess an [25]. Eine Beispielanwendung hierfür ist SonarQube. Außerdem muss der Einsatz der Programmiersprache gezielt evaluiert werden. Programmiersprachen wie C oder C++ erlauben einen freien Zugriff auf den Speicher, wodurch Speicherlecks seitens der Programmiersprache nicht abgefangen werden können. Auch dynamische Programmiersprachen wie Python oder Ruby ermöglichen es Entwicklern, leicht sicherheitsrelevante Fehler einzubauen, da sie beim Programmieren keine direkte Sicht auf den Typ der Daten haben. Um Risiken präventiv zu minimieren, muss auch bei der Auswahl von Bibliotheksfunktionen Sorgfalt walten. Im Dezember 2021 wurde bei der Apache log4j Bibliothek eine Sicherheitslücke entdeckt, die es Angreifern ermöglichte, fremden Code auf dem Computer auszuführen [13]. Da log4j frei verfügbar ist und in vielen Applikationen für Logging-Zwecke genutzt wird, bestand und besteht immer noch die Gefahr, dass Angreifer diese Bibliothek kompromittieren und somit die Sicherheit der Anwendung gefährden. Es ist daher wichtig, bei der Integration von Third-Party-Bibliotheken in den bestehenden Code Sicherheitsprobleme zu berücksichtigen. Zudem sollte bei auftretenden Fehlern oder unvorhergesehenen Systemzuständen Ausnahme- und Fehlerbehandlung eingesetzt werden, um ein Versagen des Systems zu vermeiden. Dabei ist es wichtig, interne Zustände nicht preiszugeben und dem Nutzer nur über Fehlercodes mitzuteilen, welches Problem besteht. Durch die Preisgabe interner Schwachstellen könnten potenzielle Angreifer mehr Informationen erhalten, wie das System gehackt werden kann [77].

Testingphase

Die Testingphase eignet sich gut, um im Nachgang nicht entdeckte Sicherheitsrisiken aufzudecken und zu beheben. Security Testing ist auch ein nützliches Werkzeug für Organisationen, die in den anderen Phasen keinen Fokus auf Sicherheit legen [77]. Es gibt hierbei verschiedene Arten von Tests, die sich in ihrer spezifischen Domäne unterscheiden:

1. **Static Analysis Security Testing (SAST):** Bei SAST wird Bytecode, Quellcode oder Binärcode auf Sicherheitslücken überprüft, ohne dass das Programm ausgeführt wird. Dabei können sowohl bekannte Fehler mittels Textsuche als auch komplexe Kontrollpfade überprüft werden, die zu fehlerhaften Systemzuständen führen können. Der Vorteil besteht darin, dass der Code schon frühzeitig auf Fehler überprüft werden kann, ohne dass er vollständig sein muss. Allerdings können Laufzeitfehler nicht erkannt werden [77].

2. **Dynamic Analysis Security Testing (DAST):** DAST testet das Programm während der Laufzeit auf Sicherheitslücken, indem typische Angriffe auf die laufende Software automatisiert emuliert werden. Dieser Test ermöglicht auch die Überprüfung der korrekten und sicheren Interaktion zwischen allen Komponenten, da die Software mit allen Komponenten lauffähig sein muss. Insbesondere für Webapplikationen sind solche Tests sehr hilfreich, da häufig Code ausgeführt wird, der mit anderen Rechnern interagiert und mit statischen Tools nicht testbar ist. Ein Nachteil besteht jedoch darin, dass eine Laufzeitumgebung eingerichtet werden muss, auf der die Programme ausgeführt werden können, was zu längeren Testzeiten führt[77].

3. **Fuzz Parser:** Fuzz-Parser sind relevant für Software, die Parser-Komponenten enthält. Der Parser wird in einer großen Anzahl von Iterationen automatisiert auf zufälligen Daten getestet. Das Ziel dabei ist es, eine Eingabe zu finden, die einen fehlerhaften Zustand herbeiführt. Ein Nachteil ist die Einrichtung der Tests, die auf den Parser abgestimmt sein müssen[77].

4. **Penetration Testing**: Beim Penetration Testing wird versucht, ein System aktiv zu kompromittieren, um potenzielle Sicherheitslücken zu identifizieren. Hierfür werden verschiedene Werkzeuge und Tools eingesetzt. Mögliche Angriffspunkte sind beispielsweise die Netzwerkkonfiguration der Software oder Authentifizierungskomponenten wie Passwortabfragen. Dies kann in Form eines Black-Box-Tests oder eines White-Box-Tests geschehen. Beim Black-Box-Test ähnelt der Angriff einem echten Angriff und es sind keine Informationen über die Interna des Systems bekannt. Beim White-Box-Test hingegen werden gezielt Schwachstellen im System aufgrund interner Informationen

angegriffen. Nach jedem Test müssen die Ergebnisse protokolliert und ein Bericht erstellt werden, der die aufgefundenen Schwachstellen auflistet und bezüglich ihres Schweregrads einschätzt. Penetration Testing bietet den Vorteil, dass feingranulare und möglichst realistische Problemszenarien simuliert werden können. Ein Nachteil besteht jedoch darin, dass für die Durchführung der Tests viel Fachwissen erforderlich ist. Es ist daher notwendig, dass Experten für Penetration Testing in der Organisation vorhanden sind, um diese Tests durchzuführen[77].

Deploymentphase

Nach Fertigstellung und Inbetriebnahme einer Software können nicht alle Sicherheitsrisiken ausgeschlossen werden. Außerdem können Sicherheitskomponenten veralten, wenn neue Methoden oder Erkenntnisse gefunden werden, sodass sie ihre Versprechen bezüglich bestimmter Sicherheitseigenschaften nicht mehr erfüllen. Im Jahr 2006 wurde festgestellt, dass der Hash-Algorithmus MD5 nicht kollisionsfrei ist. Es war möglich, X.509-Zertifikate mit demselben Hash zu erstellen [83]. Daher ist es notwendig, die Komponenten der Software kontinuierlich zu evaluieren und zu überprüfen, ob ihre Sicherheitseigenschaften noch erfüllt werden. Moderne Softwareentwicklung umfasst jedoch nicht nur die Software als alleinigen Fokus, sondern bezieht auch die Infrastruktur in den Entwicklungsprozess mit ein. Es ist wichtig, dass die Server-Software auf dem neuesten Stand ist und nur die notwendigen Module ausgeführt werden, um Angriffsmöglichkeiten zu minimieren. Zusätzliche Komponenten wie Firewalls sollten für die Netzwerksicherheit in Betracht gezogen werden, um unerwünschten Netzverkehr zu blockieren und das System vor Ausfällen zu schützen. Es ist auch ratsam, ein Monitoring-System für Sicherheitsbedrohungen wie ein SIEM zu implementieren [60]. Dies kann helfen, risikobehafteten Netzwerkverkehr oder Systemverhalten mittels einer künstlichen Intelligenz zu erkennen und anzuzeigen. Durch Logging können vergangene Zugriffe oder Systemvorgänge protokolliert und im Nachhinein untersucht werden. Auf Basis der Erkenntnisse, die durch Monitoring und Logging gewonnen werden, können dann Maßnahmen zur Incident Response ergriffen werden. Ein Beispiel wäre die Isolierung des betroffenen Systems vom Netzwerkverkehr, um weitere Schäden bis zur Beseitigung des verdächtigen Netzwerkverkehrs zu vermeiden.

Kapitel 3

Verwandte Arbeiten

In diesem Kapitel werden vier thematisch verwandte Arbeiten vorgestellt. Die Arbeiten werden kurz zusammengefasst und die Ergebnisse präsentiert. Zudem wird ein Vergleich mit dieser Abschlussarbeit gezogen.

3.1 Tracing

Eine weitere Arbeit zum Thema Tracing ist die Studie mit dem Namen „All Eyes on Traceability: An Interview Study on Industry Practices and Eye Tracking Potential“ [2]. Im Gegensatz zu anderen Arbeiten beschäftigt sich diese Arbeit mit dem Einsatz einer automatischen und noch hypothetischen Tracinglösung in Form eines Eye-Tracking-Systems. Um Tracing mit Eye Tracking durchzuführen, könnte man die Augen des Entwicklers beim Arbeiten durch einen bildschirmbasierten Eye Tracker verfolgen, ohne den Entwickler zu beeinflussen. Es könnten Trace-Links zwischen Artefakten gesetzt werden, indem Hotspots bestimmt werden, an denen der Entwickler besonders oft hinschaut. Wenn Hotspots für zwei Artefakte zeitgleich aufgezeichnet werden, wird implizit angenommen, dass diese beiden Artefakte miteinander verbunden sind und somit ein Trace-Link erstellt wird. Dies hat den Vorteil, dass es vollautomatisch geschieht und keine Auswirkungen auf den Entwicklungsfluss hat. Außerdem können so Informationen erfasst werden, die für Trace Links relevant sind. Dies ist schwierig, wenn man nur Textverarbeitung (z.B. Information Retrieval, Deep Learning) verwendet, um einen Zusammenhang zwischen mehreren Artefakten herzustellen. Das Ziel der Arbeit bestand darin, Erkenntnisse zur Verwaltung von Anforderungen und der Anwendung von Tracing zu gewinnen. Zudem sollten die Anforderungen an eine automatische Tracinglösung ermittelt werden. Des Weiteren wurde untersucht, wie Akteure in der Softwareentwicklung Eye Tracking für Tracing empfinden und bewerten. Die Studie wurde in Form eines semistrukturierten Interviews mit 20 Akteuren aus der Softwareentwicklung durchgeführt. Es wurde festgestellt,

dass fast alle Teilnehmer textuelle Dokumentationen wie Use Cases und User Storys verwenden und Tickets zur Verwaltung von Anforderungen einsetzen. Diese Tickets werden dann in Ticket-Systemen wie Jira verwaltet. Einige Teilnehmer waren mit dem Konzept der Traceability vertraut, während andere es unterschiedlich interpretierten. Von 20 Teilnehmern sahen 15 den Aufwand für Traceability als wertvoll an. Verlinkungen zwischen Dokumentationen und Tickets sowie zwischen Tickets und Code-Artefakten wurden häufig hergestellt. Die Verwendung von Traceability umfasst hauptsächlich die Sicherstellung der Testabdeckung, das Debugging, das Aufzeigen von Abhängigkeiten und die Unterstützung bei Entscheidungen. Wünschenswert sind Verlinkungen zwischen Code und Anforderungen, Tests und Anforderungen sowie Tests und Code. Die erforderliche Genauigkeit zwischen den Artefakten variiert. Falsch positive Ergebnisse wurden jedoch als besonders problematisch aufgefasst. Eye-Tracking als Methode für das Tracing wurde positiv aufgenommen. Ein Vorteil ist die Reduzierung des manuellen Aufwands und die Ermittlung von manuell nicht erkannten Trace-Links. Bedenken gab es hinsichtlich des Datenschutzaspekts, der praktischen Durchführbarkeit sowie der Genauigkeit und Relevanz der erfassten Trace-Links. Die Studie hebt hervor, dass Eye-Tracking-Potential als mögliche Methode für das Tracing genutzt werden kann. Allerdings muss die praktische Umsetzung noch konzipiert werden.

3.2 Security

Die Arbeit „Security in the Software Development Lifecycle“ [7] beschäftigt sich mit Sicherheitspraktiken, die in der Industrie während des Softwareentwicklungszyklus verwendet werden. In einem Interview wurden 13 Entwickler zu ihren Aufgaben, Prioritäten und den für die Softwaresicherheit relevanten Tools befragt. Um einen Kontrast zur Theorie zu schaffen, wurden die besten Sicherheitspraktiken aus der Literatur extrahiert und mit den ermittelten Sicherheitspraktiken verglichen. Die zwölf besten Praktiken aus der Literatur zur Sicherheitsentwicklung von Software sind: Identifikation von Sicherheitsanforderungen, Softwaredesign mit Fokus auf Sicherheit, Threat Modelling, Anwendung von Sicherheitsstandards bei der Implementierung, Verwendung von zertifizierten Tools, Testen mit Fokus auf Sicherheit, Code-Analyse, Code-Reviews für Sicherheit, Post-Development-Testing, Sicherheit im gesamten SDLC, Verantwortlichkeit für Sicherheit aller Akteure in der Softwareentwicklung und Absicherung der Software selbst bei niedrigem Sicherheitsrisiko. Um eine Unterteilung zwischen den Softwareteams zu schaffen, wurden einzelne Entwickler als sicherheitsbewusst (Gruppe A) und sicherheitsignorierend (Gruppe B) eingestuft. Im Allgemeinen scheitern die meisten daran, Sicherheit in allen Phasen konsistent anzugehen. Gruppe

B konzentriert sich dabei nur auf eine spezifische Phase. Dies liegt daran, dass Sicherheit oft nur von einzelnen Akteuren wie Testern behandelt wird, da es an ausreichendem Personal oder anderen Ressourcen mangelt und Manager oft keine Anforderungen an die Sicherheit stellen. Viele Teams vernachlässigen insbesondere die Designphase, da sie mit Mehraufwand verbunden ist und Deadlines eingehalten werden müssen. Im Gegensatz dazu schaffen es die meisten Teams, die Software hinsichtlich Sicherheitsaspekten nach der Entwicklungsphase zu testen. Allerdings variiert die Tiefe der Tests zwischen den Teams. In Gruppe B wird zudem:

- Applikationen mit einem niedrigen Sicherheitsrisiko werden gänzlich ignoriert
- Es gibt eine starke Abhängigkeit auf Frameworks, die vorher nicht unter dem Aspekt der Sicherheit begutachtet werden, da hier das Wissen für Sicherheitsstandards bei der Implementierung fehlen.
- Keine Beachtung der Sicherheit in funktionalen Tests und fehlen an Code Analyse, da diese in der Implementierungsphase stattfinden und Sicherheit oft nur von Testern behandelt wird.
- Code Reviews fokussieren sich nur auf die Funktionalität des Codes statt der Sicherheit, da nicht genug Wissen vorhanden ist über das Beurteilen dessen.

Änderungsanreize zur Beseitigung dieser Probleme können geschaffen werden, indem externe Instanzen Standards setzen, die von den Teams befolgt werden müssen. Die Erfüllung dieser Standards wird evaluiert. Einige Entwickler haben behauptet, dass das Finden eines Sicherheitsrisikos oder sogar das Erleben eines sicherheitskritischen Vorfalls dazu geführt hat, dass Sicherheitspraktiken besser durchgeführt und ein Bewusstsein für Sicherheit geschaffen wurde.

3.3 Tracing von sicherheitsrelevanten Artefakten

Die Arbeit „Managing Security Control Assumptions using Causal Traceability“ [66] beschäftigt sich mit dem Problem von Verletzungen der Sicherheitsanforderungen von Softwaresystemen, die durch falsche Annahmen über die Betriebsumgebung verursacht werden. Als Lösung schlagen die Autoren der Arbeit einen Ansatz vor, der kausale Rückverfolgbarkeit zwischen Sicherheitskontrollen (eng. security control) und Sicherheitsanforderungen nutzt. Dies wird gemacht, um die Ursache der Verletzungen zu identifizieren und im Anschluss zu beheben. Als erster Schritt werden Kausalitätsbeziehungen zwischen Sicherheitsanforderungen und Sicherheitsspezifikationen aufgestellt, um die logischen Folgen und Abhängigkeiten

darzustellen. Aufbauend auf diesen Beziehungen werden Annahmen über die Laufzeitumgebung dokumentiert. Diese Annahmen werden im Nachhinein dann mit Logdateien abgeglichen, die tatsächliche Umgebungsvariablen aufzeichnen, um zu überprüfen, ob die Annahmen gültig sind oder verletzt wurden. Falls eine Verletzung der Annahmen vorliegt, werden alternative Sicherheitsspezifikationen vorgeschlagen, in dem Tracing auf Basis der Kausalitätsbeziehungen durchgeführt wird. Dieser Ansatz wird anhand eines Beispiels eines elektronischen Patientenakten-Systems illustriert. Mit diesem Beispiel wird demonstriert, wie der Ansatz helfen kann, Schwachstellen in den Sicherheitsspezifikationen zu erkennen und zu beheben, indem die Annahmen überprüft und die Software angepasst wird.

Eine weitere Arbeit „Leveraging Traceability to Integrate Safety Analysis Artifacts into the Software Development Process“ stellt eine mögliche Softwarelösung für die Sicherheitsaspekte eines Systems vor [1]. Risiken und Fehler in sicherheitskritischen Systeme sind fatal, da das Ausfallen der Systeme mit diversen Schäden verknüpft ist. Die Analyse von sicherheitskritischen Systemen, um diese Fehler zu verorten, ist schwierig, da das laufende System konstant Änderungen ausgesetzt ist. SACs (Security Assurance Cases) bieten eine Möglichkeit, Sicherheitsziele oder Aussagen über die Sicherheit des Systems in Simulationsergebnisse, Testfällen oder formalen Beweisen aufzuteilen. Das Problem mit SACs ist aber, dass es keinen wirklichen Mechanismus gibt, um diese zu effektiv zu verwalten und in den Softwareentwicklungsprozess dynamisch einzubinden. Es stellt sich bei Änderungen am System die Frage, aus welchen Gründen das System sich geändert hat, welche Risiken damit verbunden sind und wie diese Änderungen die Sicherheit verändert. Die Autoren stellen eine Lösung in Form einer Softwarelösung vor. Der Zustand der Artefakte wird durch eine Anbindung an bestehende Softwarelösungen wie Jira oder DOORS verfolgt. Die Tracelinks zwischen den Artefakten und Änderungen dieser werden innerhalb der Softwarelösung mitverfolgt und auch visualisiert, mitsamt der SACs, die auf diesen aufbauen und angepasst werden müssen. Zudem werden Design rationale einer Änderung und auch die Gründe dessen bei einer Änderung am System erhoben. Dadurch kann zudem sichergestellt werden, dass Nutzer von der Software erfahren, wie eine Änderung die Sicherheit des Systems beeinflusst und ob zusätzliche Änderungen notwendig sind. Um die Softwarelösung zu demonstrieren, wurde eine neue Funktionalität zu einem Drohnensystem für Notfalleinsätze implementiert, wobei die Software in den Entwicklungsprozess mitintegriert wurde.

3.4 Abgrenzung zu dieser Arbeit

Die genannten Arbeiten konzentrieren sich jeweils auf Traceability, sichere Softwareentwicklung und das Tracing von sicherheitsrelevanten Artefakten. Ähnlich wie in dieser Arbeit wurden in den ersten beiden Arbeiten eine Interviewstudie durchgeführt. Diese betrachtet das Thema Traceability und sichere Softwareentwicklung aber isoliert und betrachtet nur einen Teilaspekt (Eye-Tracking und wie gut sichere Softwareentwicklung umgesetzt wird). Die letzteren beiden Arbeiten zu der Traceability von sicherheitsrelevanten Artefakten ist zwar thematisch ähnlich, ist aber wie in den anderen zwei Arbeiten thematisch stark eingegrenzt und stellt eine konkrete Lösung auf ein spezifisches dar. Diese Abschlussarbeit verbindet zwei getrennte Themenbereiche, indem sie Traceability-Strategien und -Verfahren untersucht, die sich auf Artefakte beschränken, die sicherheitsrelevante Konzepte umsetzen. Ähnlich wie in der Studie „An Empirical Study on Project-Specific Traceability Strategies“ wird die bisherige Forschung in diesem Themenfeld mit Fokus auf die Industrie erweitert. Das Hauptziel dieser Arbeit besteht darin, diese Wissenslücke zu schließen. Die Erwartungshaltung besteht darin, neue Erkenntnisse im Vergleich zu bestehenden Traceability-Studien in der Industrie zu gewinnen. Dies ist aufgrund des Sicherheitsaspekts der Artefakte möglich und kann zu differenzierteren Handlungsweisen bei der Verfolgung solcher Artefakte führen.

Kapitel 4

Design des Interviews

In diesem Kapitel wird das Design der Interviewstudie vorgestellt, die zur Beantwortung der Forschungsfragen durchgeführt wurde. Das Interviewformat erfordert eine genaue Struktur und Planung, um gezielt nur Antworten zu erhalten, die zur Beantwortung der Forschungsfragen dienen. Die genaue Konzeption der Methodik ist wichtig, um das Interview reproduzierbar und schlüssig zu gestalten.

4.1 Auswahl der Interviewform

Das Interview als Gesprächsformat ist eine zielgerichtete mündliche Kommunikation, die zwischen mehreren Personen, dem Moderator (oder mehreren Moderatoren) und den Interviewten (oder mehreren Interviewten) stattfindet, wo der Interviewte seine Einstellungen, Erfahrungen und Verhaltensweisen dem Moderator mitteilt [76]. Die Strukturierung des Interviews und der Grad der Kontrolle, den der Moderator über den Gesprächsfluss ausüben kann, hängen vom Format ab, in dem das Interview entworfen wurde. Es gibt im Wesentlichen drei Formate, die sich in ihrem Grad an Flexibilität unterscheiden:

- **Strukturierte Interviews:** Strukturierte Interviews sind standardisierte Interviews, bei denen alle Kandidaten dieselben vorab festgelegten Fragen in einer bestimmten Reihenfolge beantworten müssen. Zudem ist die Reihenfolge der Fragen auch in einer festgelegten Reihenfolge [76]. Es ist hierbei auch möglich, die Antwortmöglichkeit statisch vorzugeben [80]. Daher ist ein strukturiertes Interview einer Befragung sehr ähnlich und unterscheidet sich hauptsächlich dadurch, dass eine Person die Fragen mündlich an den Befragten weitergibt.
- **Semistrukturierte Interviews:** Semistrukturierte Interviews beinhalten, wie auch strukturierte Interviews, vordefinierte Fragen. Allerdings obliegt es im Rahmen semistrukturierter Interviews dem Modera-

tor, welche Fragen gestellt werden und in welcher Reihenfolge, abhängig von den Antworten des Befragten. Somit können die Fragen an die Antworten des Interviewten angepasst werden. Dadurch unterscheiden sich die gestellten Fragen von Interview zu Interview, obwohl der Pool an möglichen Fragen gleich bleibt. [80]

- **Unstrukturierte Interviews:** In unstrukturierten Interviews gibt es keinen Fragenkatalog, sondern nur festgelegte Themen. Der Gesprächsverlauf orientiert sich vollständig an den Fragen des Moderators und idealerweise auch an den Antworten des Befragten, um einen kohärenten Interviewfluss zu gewährleisten. [80]. Somit ähnelt diese Art von Interview einem offenen Gespräch und verläuft folglich auch unterschiedlich zwischen verschiedenen Interviewpartnern.

Das strukturierte Interviewformat ist für die Beantwortung unserer Forschungsfragen nicht geeignet. Wir haben nicht die Möglichkeit, gezielt weitere Fragen zu stellen, wenn der Interviewpartner eine Frage nicht ausreichend beantwortet hat. Dies ist für unsere Fragestellungen wichtig, da das Thema Tracing es erlaubt, auf verschiedene Arten ins Detail zu gehen. Zudem kann es passieren, dass der Interviewende keine Erfahrung mit Tracing hat oder keine Probleme mit dem bestehenden Tracing identifizieren kann. Daher ist eine gewisse Entscheidungsfreiheit bei der Auswahl der Fragen erforderlich. Obwohl ein unstrukturiertes Format grundsätzlich für die Beantwortung der Forschungsfragen geeignet ist, gibt es in unserem Kontext dennoch diverse Probleme. Insbesondere das breite Themenfeld der Traceability kann dazu führen, dass Antworten zu detailliert ausfallen. Es könnte passieren, dass ein Aspekt einer Forschungsfrage im Detail beantwortet wird, während die Antworten auf andere Fragen aufgrund der zeitlichen Einschränkungen des Interviews zu kurz ausfallen. Die vollständig flexible Natur des unstrukturierten Interviews erschwert auch die spätere Analyse der Ergebnisse, da unterschiedliche Gesprächsverläufe dazu führen können, dass kein einheitlicher Nenner gefunden wird. Zudem kann es vorkommen, dass der Moderator Fehler bei der Formulierung der Fragen macht oder bestimmte Aspekte aufgrund fehlender gezielter Fragen nicht anspricht, wodurch die Qualität der Ergebnisse beeinträchtigt wird. Das semistrukturierte Format ist für Interviews optimal, da ein Fragenkatalog besteht, der potenziell für alle Interviewpartner gleichermaßen gilt. Dennoch haben wir die Freiheit, je nach Inhalt der Antwort zu entscheiden, ob wir eine Frage stellen oder nicht. Die Fragen sind so konzipiert, dass sie Ergebnisoffenheit zulassen. Die Mehrheit der Antworten ist nicht auf Ja oder Nein festgelegt.

4.2 Interviewleitfaden

4.2.1 Grundlegendes

Der Interviewleitfaden enthält die Fragen, die während des Interviews gestellt werden. Um den Gesprächsverlauf zu erleichtern und den Einschränkungen des semistrukturierten Interviewformats gerecht zu werden, wird die Reihenfolge der Fragen festgelegt. Falls eine Antwort nicht zufriedenstellend ist, werden alternative Fragen bestimmt und optionale Fragen zu den Hauptfragen hinzugefügt. Der gesamte Interviewleitfaden ist im Detail im Anhang A nachlesbar. Zunächst muss die Dauer des Interviews festgelegt werden. Eine Mindestlänge kann nicht bestimmt werden, da der Interviewpartner nur so viel sagen kann, wie ihm oder ihr zum Zeitpunkt des Interviews möglich ist. Um eine reibungslose zeitliche Organisation des Interviews zu gewährleisten und die Nachbereitung nicht zu zeitaufwendig zu gestalten, wird die Maximaldauer auf eine Stunde begrenzt. Dies kann erreicht werden, indem die Anzahl der Fragen an die Zeitbeschränkung angepasst wird. Um die Dauer des Interviews zu validieren, sollten Testläufe durchgeführt werden und gegebenenfalls der Leitfaden angepasst werden. Für die Validierung dieses Interviewformats wurde ein Testinterview mit einem wissenschaftlichen Mitarbeiter durchgeführt, der die jeweiligen Kriterien für einen möglichen Interviewpartner erfüllt. Nach der Durchführung der Testläufe stellte sich heraus, dass es mit dem Interviewleitfaden möglich ist, ein Interview in 35 Minuten durchzuführen, was weit unter der maximalen Zeit von einer Stunde liegt. Dieser Zeitwert ist ein guter Indikator dafür, dass das Interview innerhalb des angepeilten Zeitrahmens durchgeführt werden kann, selbst bei gesprächigen Partnern. Darüber hinaus wurde der Interviewleitfaden inhaltlich angepasst, da einige Fragen nicht alle Antwortmöglichkeiten abdeckten. Es war möglich, dieselbe Antwort wiederholt auf mehrere Fragen zu geben.

4.2.2 Einleitung

Vor der Einleitung wird darauf hingewiesen, dass die Interviewaufzeichnung beginnt. Die mündliche Zustimmung wird nach dem Start der Aufzeichnung erneut eingeholt, um sie in der Transkription zu protokollieren. Wenn keine Zustimmung erteilt wird, wird das Interview abgebrochen und die Aufnahme gelöscht. Zu Beginn des Interviews wird der Interviewteilnehmer begrüßt. Im Anschluss stellt sich der Moderator vor, indem er seinen Namen und sein Studium nennt. Anschließend wird der Titel der Arbeit genannt und das Thema sowie das Ziel des Interviews erläutert. Bei der Erläuterung des Themas wird sichergestellt, dass der Interviewpartner die Semantik von Tracing und der „Sicherheitsrelevanz“ von Artefakten versteht. Dazu wird er aufgefordert, diese Begriffe zu nennen und bei Unklarheiten aufgeklärt. Ein unvollständiges Verständnis könnte zu falschen Antworten aus Unwissenheit

führen. Zuletzt wird der grobe Ablauf des Interviews skizziert und die Rahmenbedingungen mündlich zusammengetragen. Es wird darauf geachtet, dass die Freiwilligkeit, die Aufzeichnung des Interviews und die Möglichkeit, Fragen nicht zu beantworten, erwähnt werden. Die Einleitung hat die Funktion sicherzustellen, dass der Interviewpartner das Thema erneut in Erinnerung gerufen bekommt und das Interview grob einordnen kann. Zudem soll er mit den formalen Rahmenbedingungen vertraut gemacht werden, um spätere Verzögerungen aufgrund von Missverständnissen vorzubeugen.

4.2.3 Soziodemografische Fragen

Zu Beginn des Interviews werden zunächst soziodemografische Informationen des Interviewpartners erfragt. Dazu zählen die generelle Berufserfahrung sowie die Berufserfahrung beim aktuellen Arbeitgeber. Die Erfahrungsebene dient der Einschätzung, ob bestimmte Aspekte, wie beispielsweise Probleme im Tracing, in Korrelation mit der Anzahl der Berufsjahre anders bewertet werden. Verschiedene Rollen in der Softwareentwicklung haben unterschiedliche Problemdomänen. Daher könnte eine weitere Erkenntnis aus den verschiedenen Rollen und dem rollenspezifischen Tracing gewonnen werden. Zuletzt kann der Sektor, in dem das Unternehmen tätig ist, relevant für unsere Analyse sein. Der Interviewpartner wird gebeten, diese Fragen kurz zu beantworten, da offene Antworten keinen größeren Mehrwert für unsere Analyse bieten.

4.2.4 Hauptteil

Der Hauptteil beantwortet gezielt die Forschungsfragen. Jede Frage dient dazu, einen Teilaspekt der jeweiligen Forschungsfrage zu klären. Die Fragen können thematisch innerhalb der beiden Hauptfragen mit jeweils drei Unterfragen bei der zweiten Hauptfrage eingeordnet werden. Wichtig ist eine klare Strukturierung, beginnend mit Fragen, die sich speziell auf die Definition der sicherheitsrelevanten Artefakte des Unternehmens beziehen, gefolgt von Fragen zur Nachvollziehbarkeit dieser Artefakte. Zu Beginn des Interviews sollten sowohl der Interviewte als auch der Moderator sich ein Bild von den sicherheitsrelevanten Artefakten im jeweiligen Unternehmen/Team machen. Dies ist wichtig, um sicherzustellen, dass der Fokus bei den Fragen zum Tracing auf diesen Artefakten liegt und nicht auf dem Tracing ohne Berücksichtigung der Sicherheitsrelevanz. Die erste Frage zielt darauf ab, dass der Interviewte Beispiele für sicherheitsrelevante Artefakte nennt. Als Einstiegsfrage ist diese Frage besser geeignet als die Definition der Eigenschaften dieser Artefakte, da man im Alltag eine größere Verbindung zu spezifischen Beispielen hat als zur Verallgemeinerung dieser Artefakte. Nachdem der Interviewpartner diese Frage beantwortet hat, wird die Frage ergänzt, indem jeweils gefragt wird, ob eine klare Identifizierung dieser

Artefakte auch auf Projektebene gilt, z.B. in Form einer textuellen Kennzeichnung innerhalb eines Ticketsystems. Je nach Antwort wird entweder gefragt, warum dies nicht getan wird oder, falls die Antwort positiv ausfällt, ob diese Artefakte gesondert behandelt werden. Abschließend wird nach einer allgemeinen Differenzierung von sicherheitsrelevanten und nicht-sicherheitsrelevanten Artefakten seitens des Teams/Unternehmens gefragt. Falls auf Prozessebene keine Identifizierung der Artefakte stattfindet, kann der Interviewpartner die Frage aus seiner eigenen Einschätzung heraus beantworten. Im Anschluss folgen Fragen zum Tracing. Es wird gefragt, ob sicherheitsrelevante Artefakte innerhalb des Unternehmens/Teams getraced werden und falls ja, ob dies differenzierter durchgeführt wird als das Tracing von generischen Artefakten. Je nachdem, ob eine Differenzierung beim Tracing vorliegt, werden die folgenden Fragen beantwortet. Dabei werden die jeweiligen Besonderheiten des Tracings hervorgehoben. Bei einer negativen Antwort werden mögliche Anpassungsforderungen geäußert. Beispielsweise sollte bei der nächsten Frage nach den Vorteilen des Tracings der Fokus auf dem Tracing von sicherheitsrelevanten Artefakten liegen oder die Vorteile des Tracings von generischen Artefakten beschrieben werden, wobei Anpassungsmöglichkeiten für sicherheitsrelevante Artefakte gewünscht sind. Im Anschluss folgen Fragen zu Methoden und Prozessen im Tracing sowie zu Problemen und Herausforderungen. Die Fragen sind strukturiert, beginnend mit einer grundlegenden Frage zu einem Aspekt, gefolgt von 1 bis 3 detaillierteren Fragen, je nach Tiefe der Antwort. Dieses Muster findet sich auch in den weiteren Fragen wieder.

4.2.5 Schluss

Im Anschluss wird der Interviewpartner gefragt, ob ihm im Nachhinein noch Erkenntnisse zu den Fragen eingefallen sind. Es kann nämlich passieren, dass während des Interviews bestimmte Aspekte nicht direkt in den Sinn kommen. Außerdem wird nach weiteren Aspekten gefragt, die im Interview nicht ausführlich behandelt wurden, da aus zeitlichen Gründen nicht alles im Detail besprochen werden kann. Abschließend zum Interview wird für die Teilnahme dessen gedankt, die Audioaufnahme beendet und der Teilnehmer verabschiedet.

4.3 Interviewteilnehmer

4.3.1 Zielgruppe der Interviewteilnehmer

Die Auswahl der Interviewteilnehmer hat einen starken Einfluss auf die Qualität des Interviews, da die Antworten von ihrem Wissen und ihren persönlichen Erfahrungen abhängen. Es ist daher wichtig, gezielt Personen auszuwählen, die bestimmte Kriterien erfüllen. Da die Interviewstudie den

Zustand des Tracings von sicherheitsrelevanten Artefakten in der Industrie untersucht, ist es ein notwendiges Kriterium für den Teilnehmer, zum Zeitpunkt des Interviews in einem Unternehmen zu arbeiten. Die Rolle der Person in diesem Unternehmen ist flexibel zu verorten, sobald sie in jeglicher Art und Weise direkt an Softwareprojekten mitarbeitet und somit in der Lage ist, Aussagen über das Tracing innerhalb eines oder mehrerer Softwareprojekte zu machen. Unser Fragenkatalog schließt auch die Nichtexistenz von Tracing sicherheitsrelevanter Artefakte ein. Es ist irrelevant, ob der Entwickler oder sein Team Tracing solcher Artefakte aktiv verwendet oder nicht. Es ist nur wichtig, dass der Interviewpartner in einem Unternehmen arbeitet, in dem generell Tracing betrieben wird, unabhängig vom Artefakt. Somit kann er auch die Nichtexistenz von spezialisierterem Tracing von sicherheitsrelevanten Artefakten bewerten und in Verbindung mit dem aktuellen Stand der Entwicklungsprozesse bringen. Eine weitere Bedingung ist, dass der Interviewpartner mindestens 2 Jahre Industrieerfahrung hat. Es ist möglich, mehrere Interviewpartner aus einem Unternehmen zu befragen. Allerdings ist es vor der Durchführung des Interviews anzunehmen, dass Personen, die in derselben Rolle in einem Unternehmen tätig sind, ähnliche oder zusammenfassende Antworten auf unsere Fragen geben werden, verglichen mit den Antworten ihrer Arbeitskollegen. Dies liegt daran, dass die Art und Weise des Tracings und die Bestimmung der sicherheitsrelevanten Artefakte nicht auf individueller Basis erfolgt, sondern auf der Ebene des jeweiligen Teams, da die Artefakte von mehreren Akteuren in der Softwareentwicklung bearbeitet werden. Deswegen gilt als weitere Bedingung, dass bei mehreren Interviewteilnehmern aus einem Unternehmen jeweils unterschiedliche Rollen in einem Unternehmen haben müssen, da davon auszugehen ist, dass diese eine andere Perspektive und Sichtweise durch die unterschiedlichen Rollenverteilungen innerhalb der Softwareentwicklung haben werden.

4.3.2 Teilnehmerakquise

Es wurden zwei Methoden angewendet, um Interviewteilnehmer zu finden: Die Interviewstudie wurde über ein soziales Netzwerk online publik gemacht und es wurden persönliche Kontakte genutzt, die für das Interview qualifiziert sind. Die erste Methode wurde durchgeführt, indem auf dem sozialen Netzwerk LinkedIn Werbung für das Interview gemacht wurde. Dabei wurde ein Text erstellt, der die Ziele der Arbeit beschreibt und die Rahmenbedingungen klärt. Um ein größeres Publikum zu erreichen, wurde die Werbung im LinkedIn-Netzwerk des Abschlussarbeitsbetreuers geteilt. Diese Methode war jedoch nicht erfolgreich, da keine Antworten auf die Werbung geäußert wurden. Ein Teilen der Werbung in anderen sozialen Netzwerken, wie Reddit, wurde aufgrund der Anonymität des Netzwerks und der damit einhergehenden Unsicherheit bezüglich der Professionalität der

möglichen Interviewteilnehmer vermieden. Somit konnte der Erfolg nur durch die Bestimmung der Interviewpartner über persönliche Kontakte erreicht werden. Es wurden acht Teilnehmer befragt, da die Interviewlänge sich auf ungefähr 1 Stunde beläuft und diese bei vielen Interviewteilpartner zeitlich schwer nachzubereiten ist. Die jeweiligen Interviewpartner wurden individuell unterschiedlich angesprochen. Es wurde also kein einheitlicher Text oder Dokument benutzt, um für das Interview zu werben. Dies hat sich als eine gute Strategie erwiesen, da die persönliche Nähe zu den Teilnehmer und auch die Qualifizierung der Teilnehmer unterschiedlich war und somit die Anwerbung dynamisch angepasst werden konnte. Die initial angesprochenen Teilnehmer haben weitere Personen vorgeschlagen, die ihrer Meinung nach für das Interview qualifiziert genug wären. Einige angefragte Personen zeigten zunächst Interesse an dem Interview, jedoch wurde vor der Durchführung eine Absage erteilt oder es wurde nicht mehr auf Folgeanfragen geantwortet. Der Grund für die Absage war das Fehlen persönlicher Kenntnisse über das Thema Traceability. Außerdem besteht im Team des jeweiligen Teilnehmers die Richtlinie, dass Informationen über die Projekte des Teams nicht veröffentlicht werden können, selbst unter Einhaltung der DSGVO und Anonymisierung der Transkripte. Daher konnten die fachlich qualifizierten Arbeitskollegen des Teilnehmers aufgrund dieser Richtlinien nicht berücksichtigt werden. Es ist anzunehmen, dass diese Gründe auch für den Misserfolg der Online-Recherche gelten, einschließlich der fehlenden persönlichen Nähe des Teilnehmers zu den Personen im Netzwerk. Vor der Durchführung des Interviews wurden die Interviewteilnehmer dazu gebeten, die Einwilligungserklärung in A zu lesen und zu unterschreiben. Mit der Unterschrift haben die Teilnehmer eingewilligt, dass sie am Interview freiwillig teilnehmen und das Interview aufgezeichnet wird, aber das Interview jederzeit abgebrochen werden kann und das Einverständnis im Generellen auch widerrufen werden kann. Zudem wurde zugestimmt, dass die Interviewaufzeichnung im Nachgang anonymisiert wird und diese Daten für weitere wissenschaftliche Arbeiten genutzt werden kann.

4.4 Durchführung des Interviews

Die Interviews fanden im Februar 2024 mit den Interviewteilnehmern statt und wurden anhand eines vorher erstellten Leitfadens durchgeführt. Der Leitfaden A.2 ermöglichte es, optionale Fragen zu stellen oder zu überspringen und bei Bedarf nähere Nachfragen zu stellen, um ausreichend Informationen zu erhalten. Da die Kontakte in ganz Deutschland verteilt arbeiten und nicht vor Ort sind, wurde das Interview ausschließlich online durchgeführt. Jeder Teilnehmer konnte vor dem Interview auswählen, welche Audiokonferenzsoftware er nutzen möchte. Dadurch war sichergestellt, dass er bereits mit der Bedienung der ausgewählten Softwarelösung vertraut war.

Es wurde Google Meet, Cisco WebEx und Microsoft Teams ausgewählt. Bei zwei Teilnehmern gab es hierbei keine Präferenz, weswegen die Open-Source Lösung Big Blue Button benutzt wurde. Die Interviews wurden lokal mit der Software OBS aufgezeichnet, um sicherzustellen, dass kein unbefugter Zugriff durch Dritte auf die Aufzeichnungen erfolgt, wenn diese zunächst auf Server hochgeladen werden. In einigen Interviews gab es Verbindungseinbrüche und die Beeinträchtigung des Interviews seitens dritter Personen, die jedoch durch die Verwendung von OBS behoben werden konnten, indem die Aufnahme gestoppt und nach der Möglichkeit, das Gespräch wieder aufzunehmen, wieder gestartet wurde.

4.5 Analyse der Interviews

4.5.1 Nachbereitung der Interviews

Die Verschriftlichung der Interviews ist eine notwendige Voraussetzung für eine methodische Analyse. Der manuelle Transkriptionsprozess ist jedoch zeitintensiv und anfällig für Fehler. Eine Softwarelösung auf Basis von maschinellem Lernen könnte hier Abhilfe schaffen und die Arbeit erleichtern. Zum Zeitpunkt der Arbeit stehen verschiedene Tools zur Verfügung, die sich hinsichtlich ihrer Präzision (Fehlerrate in transkribierten Wörtern), Kosten und auch der Lokalität der Ausführung unterscheiden [69, 5, 57]. Eine Softwarelösung, die in allen drei Aspekten nach dem derzeitigen Stand der Technologie optimale Ergebnisse liefert, ist Whisper [74, 68]. Whisper ist eine multifunktionale Künstliche Intelligenz, die auf Sprachverarbeitung spezialisiert ist und einen Transformer-Ansatz verfolgt. Whisper unterstützt das Transkribieren, Übersetzen, die Erkennung der Sprache sowie Füllwörter (wie Ähm, Hmm, oh, ah etc.) in mehreren Sprachen, darunter auch in der deutschen Sprache. Das Tool wird auf Linux installiert und über die Kommandozeile bedient. Um ein kompatibles Audioformat für Whisper zu haben, wurden die MP4-Dateien zunächst in ein MP3-Format umgewandelt. Diese können der Whisper-KI als Eingabe übergeben werden. Die Transkription selber kann dann jeweils in unterschiedlichen Dateiformaten ausgegeben werden. Für die Transkription wird das srt-Dateiformat verwendet, da es eine klare und gut lesbare Unterteilung der gesprochenen Textblöcke bietet und zudem den Vorteil hat, dass bei der Wiedergabe der Transkriptionsaudios die textuelle Transkription in Form von Untertiteln eingebunden werden kann. Die KI wird dabei lokal auf der Grafikkarte ausgeführt, die Ausführungszeit betrug hierbei ungefähr fünf Minuten. Für die Transkription wird das zweitgrößte Modell „large-v2“ verwendet, da dieses im Allgemeinen eine sehr niedrige Fehlerrate bezüglich richtig transkribierter Wörter aufweist. Zudem wurde bei einem Test mit dem größten Modell „large-v3“ festgestellt, dass dieses dazu neigt, Halluzinationen zu produzieren und die Fehlerrate im Kontrast zu „large-

v2“ nicht besser ist, sondern schlechter. In der Praxis hat sich Whisper als äußerst effizientes Tool zur Beschleunigung der Transkription erwiesen. Die KI teilt das gesprochene Wort in Segmente auf und annotiert diese mit Zeitstempeln. Dies erfolgt durch die Unterscheidung von gesprochenem und nicht gesprochenem Wort (auch als „Voice Activity Detection“ bezeichnet), wobei das Modell die Segmente so lange setzt, bis das gesprochene Wort endet und somit ein Zeitslot gesetzt wird. Obgleich die Fehlerrate als äußerst gering zu bezeichnen ist, musste dennoch eine Korrektur durchgeführt werden. Die Audioaufnahme wurde angehört und parallel dazu, sofern Fehler festgestellt wurden, die Korrektur durchgeführt. Der wesentliche Zeitaufwand bestand dabei in der Zuordnung des Gesprochenen zu Interviewpartner und Moderator, da die Sprecherdiarisierung nicht zu den Funktionalitäten von Whisper zählt. WhisperX [11] stellt eine Modifikation von Whisper dar, welche die Funktionalität der Sprechererkennung unterstützt. Zudem weist WhisperX im Vergleich zu Whisper schnellere Ausführungszeiten auf. Sätze werden mit WhisperX so transkribiert, dass sie mit Sprecherkennzeichen annotiert werden. Gesprochenes wird im Verhältnis zum jeweiligen Sprecher transkribiert. Aus diesem Grund wurden sechs von acht Transkripten mit WhisperX transkribiert, was den Zeitaufwand nochmals reduzierte. Im Rahmen der Korrektur der schriftlichen Transkription wurde darauf geachtet, dass Teile des Transkripts so anonymisiert werden, dass kein Rückschluss auf den Interviewpartner im Nachhinein möglich ist. Bei der Zuweisung des Gesprochenen auf die Interviewteilnehmer wurde der Interviewpartner und der Moderator mit dem Wort „SPRECHER_“ und einer Nummer gekennzeichnet. Der Moderator wurde mit der Nummer 0 gekennzeichnet, während die Interviewteilnehmer in der Reihenfolge der Interviewdurchführung nummeriert wurden (z.B. 3. Interviewpartner erhält die Kennzeichnung „SPRECHER_3“). Namen, Ortschaften und andere Firmen, die in Verbindung mit der Person oder dem Unternehmen, mit dem die Person assoziiert werden kann, wurden durch Platzhalter ersetzt, die lediglich eine Beschreibung der jeweiligen Kategorie darstellen. Ein Beispiel hierfür ist die Nennung einer Ortschaft, die durch das Wort „Ortschaft“ ersetzt wird. Um diese Platzhalterwörter kenntlich zu machen, wurden die Platzhalter in Großbuchstaben aufgeschrieben. Zudem gab es einige Wörter und auch Satzteile in dem Audio, die nach mehrmaligem Anhören nicht entziffert werden können, da die Qualität niedrig war oder es zu einem technischen Aussetzer in der jeweiligen Stelle kam. Der Inhalt wird jedoch nur marginal beeinflusst, sodass die Analyse davon nicht betroffen ist. Aus den umliegenden Satzteilen lässt sich der Kontext dennoch ableiten, sodass die unverständlichen Wörter mit einem Platzhalterwort gekennzeichnet wurden.

4.5.2 Methodiken der Analyse

Eine geeignete qualitative Analysemethode ist bedeutsam für die Qualität der Analyseergebnisse für das Interview, da ein willkürlicher Ansatz schwer nachzuvollziehen ist und zudem durch die Unstrukturiertheit schnell das Gesamtbild außer Acht geraten kann. Eine denkbare Methode, die bei der Recherche für geeignete qualitative Methoden in Betracht gezogen wurde, ist hierbei der Grounded Theory Ansatz [33]. Der Ansatz bietet sich aber für unser Themenfeld im Spezifischen nicht an. Die Grundidee des Grounded Theory Ansatz, dass die Daten so ausgewertet werden, dass im Prozess der Auswertung Theorien basierend auf den Daten generiert werden [84]. Die Grundidee, dass man auf der Basis der Daten Theorien erstellt, würde mit unseren Zielen ein Modell zu erstellen, im Einklang sein, da Theorien hierbei ein größeres Fundament für unser Modell darstellen können. Aber die Anzahl von acht Interviews ist hierbei nicht förderlich, für die Theoriebildung, da nicht gewährleistet werden kann, dass die Daten heterogen genug sind, bedingt dadurch, dass wir Interviewpartner haben, die in demselben Unternehmen oder in ähnlichen Arbeitsdomänen arbeiten. In der Literatur wird eine Anzahl von 20 bis 30 Experteninterviews als optimal für die Grounded Theory Analyse gesehen [20], womit unsere Anzahl an Experteninterviews weit unterhalb der empfohlenen Mindestgrenze liegt. Außerdem gibt es das Problem, dass bei der Traceability und auch dem Themenfeld der sicherheitsrelevanten Artefakte eine diverse Bandbreite an Herangehensweisen und Auffassungen seitens der Industriepraktiker bestehen. Eine Generalisierung, wie es bei der Theoriefindung in dem Grounded Theory Ansatz passiert, ist nicht zielführend. Der Aspekt, der für die Gegebenheiten der Abschlussarbeit aber der größte Nachteil ist, ist das Konzept des theoretischen Samplings innerhalb der Grounded Theory [41]. Diese erfordert, dass der Analyseprozess in Abstimmung mit dem Interviewprozess in Abhängigkeit mit der Qualität und der Tiefe der Konzepte, die sich aus der Analyse erschließen, abläuft. Da wir alle Interviews vor der qualitativen Analyse durchgeführt haben, aufgrund der Einschränkung, die wir bei der Teilnehmersuche haben, kann dieser Teil der Grounded Theory nicht erfüllt werden, womit eine der Methodik gerechte Analyse nicht durchgeführt werden kann. Der zweite Ansatz, der untersucht wurde und Einsatz in dieser Abschlussarbeit findet, ist die qualitative Inhaltsanalyse. Im Speziellen benutzen wir hierbei die qualitative Inhaltsanalyse nach Mayring [58]. Die qualitative Inhaltsanalyse nach Mayring erlaubt es, Daten auf drei verschiedene Arten und Weisen zu untersuchen:

- **Zusammenfassende Inhaltsanalyse:** Diese Methode reduziert das Material auf seine wesentlichen Inhalte, sodass ein übersichtlicher Korpus entsteht, der die grundlegenden Informationen des Ausgangsmaterials beibehält. Dies wird durch Abstraktion erreicht [58].

- **Erklärende Inhaltsanalyse:** Hierbei werden zusätzliche Informationen herangezogen, um unklare Textstellen im Material zu erläutern oder besser zu verstehen. Dies kann durch Hinzufügen weiterer Quellen oder durch detaillierte Erläuterungen geschehen [58].

- **Strukturierende Inhaltsanalyse:** Ziel dieser Methode ist es, spezifische Aspekte im Material herauszuarbeiten, einen Querschnitt durch das Material zu erstellen oder es nach bestimmten Kriterien zu bewerten. Dabei wird eine Struktur entwickelt, die auf vorher festgelegten Ordnungskriterien basiert [58].

Die jeweiligen Methoden unterscheiden sich in ihren Vorgehensweisen zur Filterung und Annotation von Informationen. Da unsere Interviews einen beträchtlichen Umfang aufweisen und eine Vielzahl an gesprochenem Material für ein wesentliches Konzept enthalten, wenden wir in der Interviewanalyse die zusammenfassende Inhaltsanalyse an. Die Zusammenfassung erfolgt durch Reduktion der Absätze auf wesentliche Konzepte und deren tabellarische Darstellung in Form von Codes. In einem weiteren Schritt werden ähnliche Aspekte in einer zweiten Spalte zusammengefasst und somit zu einem Oberkonzept eingruppiert. Bei der Erstellung der Codes für die Inhaltsanalyse wird ein induktiver Ansatz verfolgt, da die jeweiligen Fragestellungen mehrheitlich in kein bestimmtes Raster im Voraus eingeordnet werden können und offen beantwortet werden müssen. Des Weiteren ist die Erstellung eines ausreichend großen Codierungsleitfadens aufgrund der unzureichenden Anzahl an Quellen für das spezifische Tracing von sicherheitsrelevanten Artefakten erschwert. Für die Vereinfachung des Analyseprozesses wird die Software MAXQDA genutzt [34]. MAXQDA bietet sich im Kontext der Analyse so an, dass Codes durch alle Transkriptionen hinweg innerhalb der Software erstellt und verwaltet werden können. Für eine Quantisierung der jeweiligen Codes bietet MAXQDA entsprechende Funktionalitäten an.

4.6 Datenschutz

Datenschutz ist ein wesentlicher Bestandteil jeder Forschungsarbeit [27], insbesondere im Themenfeld des Tracings von sicherheitsrelevanten Artefakten. Die befragten Interviewpartner müssen Informationen über interne Prozesse, Methodiken und Probleme preisgeben. Es besteht die Möglichkeit, dass die Veröffentlichung der Identität der Unternehmen ein potenzielles Sicherheitsrisiko darstellt, insbesondere im Hinblick auf den Aspekt der Sicherheit. Aus diesem Grund werden nur die Branche und Unternehmensgröße des jeweiligen Unternehmens genannt. Es werden Maßnahmen ergriffen, um die Privatsphäre der Teilnehmenden und die Sicherheit der gesammelten Daten zu gewährleisten:

- **Einwilligungserklärung:** Vor Beginn der Interviews wird von allen Teilnehmenden eine Einwilligungserklärung eingeholt. Diese Erklärung umfasst Informationen über den Zweck der Studie, die Art der gesammelten Daten und die Maßnahmen zum Schutz dieser Daten. Die Interviewteilnehmer werden darüber informiert, dass ihre Teilnahme freiwillig ist und sie das Recht haben, jederzeit ohne Angabe von Gründen aus der Studie auszusteigen. Die Erklärung ist im Anhang einzusehen A.1.

- **Anonymisierung der Daten:** Um die Anonymität der Teilnehmenden zu gewährleisten, werden alle persönlichen Identifikatoren entfernt oder mit Füllwörtern pseudonymisiert. Dies beinhaltet die Ersetzung von jeglichen Informationen, die auf die Identität des Interviewpartners oder des jeweiligen Unternehmens rückzuführen, wie z.B. Namen, Kontaktdaten von Personen, persönliche Merkmale über die Person oder auch Ortschaften.

- **Sichere Datenspeicherung:** Alle gesammelten Daten werden sicher auf institutseigenen Servern gespeichert. Somit wird der Zugriff seitens unbefugter Personen verhindert.

- **Datennutzung und -verbreitung:** Die Verwendung der Daten wurde auf die Ziele der Interviewstudie beschränkt, inklusive möglicher Referenzierung von nachfolgenden Papern, die auf den Erkenntnissen dieser Abschlussarbeit aufbauen. Alle Forschungsergebnisse werden so berichtet, dass die Identität der Teilnehmenden geschützt bleibt. Zudem werden nur Auszüge aus der Transkription innerhalb der Abschlussarbeit benutzt, da die gesamte Transkription nicht öffentlich gestellt wird.

- **Datenwiederruf:** Teilnehmer haben das Recht, Auskunft über ihre gespeicherten Daten zu erhalten, die Berichtigung falscher Daten zu verlangen und die Löschung ihrer Daten zu beantragen. Somit kann die vorher eingewilligte Einverständniserklärung widerrufen werden.

Kapitel 5

Ergebnisse

5.1 Statistische Daten über die Interviewteilnehmer

Um den Hintergrund und Legitimität der Interviewpartner einzuordnen zu können, ist es erforderlich, sich die soziodemografischen Daten der Teilnehmer anzuschauen.

5.1.1 Daten über die Interviewanfragen

Es ist von Relevanz, wie erfolgreich der Ánwerbeprozess für die Interviewteilnehmer war. Die Zahl der Personen, die über die Online LinkedIn Recherche angefragt wurden, ist schwer zu ermitteln, da es nicht möglich ist, genau zu bestimmen, wie viele die Anzeige aktiv gesehen haben und sich dann im weiteren Schritte auch komplett darüber informiert haben. Insgesamt wurden 24 Personen auf direktem und indirektem Wege angefragt. Direktheit heißt in diesem Fall, ob die Personen persönlich angefragt wurden oder über eine Drittperson. Für die direkten Anfragen wurden 14 Personen angefragt, von denen acht erfolgreich waren und zu einer weiteren Interviewdurchführung geführt haben. Vier der interviewten Personen wurden von den vier anderen Personen jeweils vorgeschlagen oder für das Interview kontaktiert. Eine Person wurde angefragt, hat aber keine Antwort auf die Anfrage zurückgegeben. Zwei Personen haben initial zugestimmt, haben aber darauffolgend keine weitere Rückmeldung auf Terminanfragen gegeben. Die restlichen drei Personen haben direkt abgelehnt. Zwei waren nach eigener Einschätzung fachlich nicht genug qualifiziert. Einer konnte aus Geheimhaltungsrichtlinien der Projekte das Interview nicht durchführen. Es wurden zudem zehn Personen indirekt nach Personen aus ihrer Bekanntschaft angefragt, die fachlich für das Interview passend wären. Acht der zehn Personen kannten keine Personen. Zwei der Personen hatten mögliche Personen in ihrer Bekanntschaft oder Team

kontaktiert, es gab hierbei wie bei den anderen Anfragen auch keine positive Resonanz. Die Daten sind anschaulich in 5.1 zu sehen.

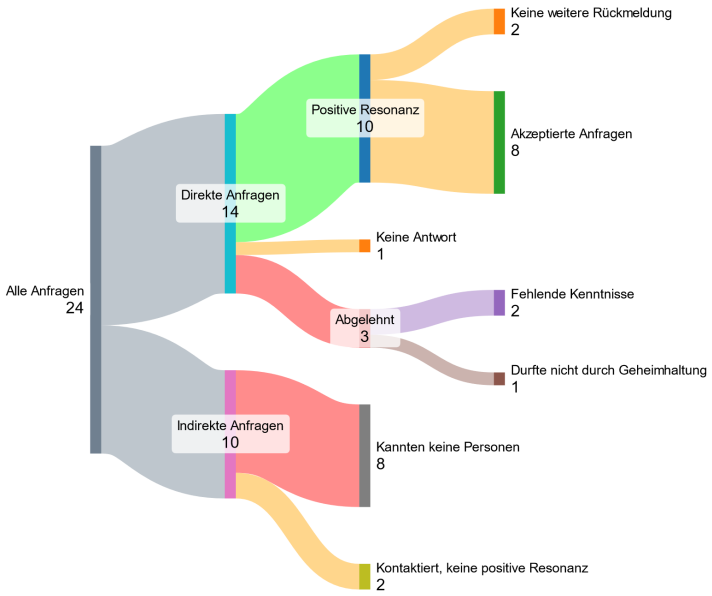


Abbildung 5.1: Daten der Interviewanfragen

5.1.2 Daten über die Interviewpartner

Um die Heterogenität der Daten zu gewährleisten, wurde darauf geachtet, dass die befragten Personen in möglichst unterschiedlichen Rollen in der Softwareentwicklung tätig sind. Wie bereits in vorherigen Abschnitten dargelegt, wurden drei soziodemografische Fragen gestellt. Die erste soziodemografische Frage zielte darauf ab, die Dauer der bisherigen Erfahrung der Teilnehmer in der Softwareentwicklung zu ermitteln. Die zweite Spalte der Tabelle 5.1 beschreibt zeilenweise die Erfahrung in 10-Jahresintervallen. Wie in der Tabelle zu sehen ist, verteilt sich die Erfahrung der Teilnehmer auf verschiedene Erfahrungslevel, was auch durch den Durchschnittswert von 20,625 Jahren und die mittlere Abweichung von 11,875 Jahren bestätigt wird. Die zweite soziodemografische Frage zielt auf die Dauer der Tätigkeit in Jahren in dem aktuellen Unternehmen der Interviewpartner, welche in der dritten Spalte der Tabelle 5.1 zu sehen ist. Hierbei ist zu berücksichtigen, dass die Werte insgesamt deutlich niedriger sind als die Werte der Arbeitserfahrung in der Softwareentwicklung insgesamt. Dies ist darauf zurückzuführen, dass der Durchschnittswert 6,5 und die mittlere Abweichung ungefähr 4,28 beträgt. Die Mehrheit der Werte liegt jedoch bei 4 oder weniger Jahren. Die letzte Frage dient der Erhebung der Rolle der Interviewpartner im Unternehmen.

Wie in der zweiten Spalte der Tabelle 5.2 ersichtlich, wurde die Bedingung erfüllt, Personen aus verschiedenen Rollen innerhalb eines Unternehmens zu befragen. Die interviewten Personen nehmen unterschiedliche Rollen ein, darunter Softwareentwickler (mit und ohne Führungsverantwortung), Functional Safety Manager, Systemingenieur, Teamleiter, Testmanager und Testingenieure. Aufgrund der geringen Anzahl an Interviewpartnern gibt es nur drei Personen, die die gleiche Rolle innehaben (Softwareentwickler). Dadurch ist gewährleistet, dass jeder Interviewpartner eine andere Rolle einnimmt und somit unterschiedliche Einblicke in die rollenspezifischen Unterschiede innerhalb des Tracings gewonnen werden können. Die interviewten Personen waren allesamt männlichen Geschlechts, da in den jeweiligen Unternehmen mehrheitlich Männer in den relevanten Tätigkeitsgruppen beschäftigt sind. In einem letzten Schritt erfolgt eine Betrachtung der Domänen der Unternehmen, in denen die Interviewpartner tätig sind. Die Interviewpartner sind auf fünf verschiedene Unternehmen verteilt, welche in Tabelle 5.3 zu erkennen sind. Wie in der dritten Spalte der Tabelle 5.2 ersichtlich, arbeiten die Interviewpartner in verschiedenen Domänen. Das Echtzeitbetriebssystem findet beispielsweise Anwendung in Fahrzeugen, in der Medizintechnik oder in Industrieanlagen. Die Gatewayelektronik wird für die Vernetzung von elektronischen Konsumgütern eingesetzt. Dennoch lässt sich feststellen, dass das Endprodukt der Unternehmen Elektronik beinhaltet oder im Falle des Echtzeitbetriebssystems in strenger Weise mit Elektronik kommunizieren muss. Dies ist darauf zurückzuführen, dass der Ausfall von Elektronik sicherheitskritische Aspekte beinhaltet, insbesondere in Bereichen wie Militärtechnik, Echtzeitbetriebssysteme und Automatisierungstechnik. Zudem spielt Traceability in der Militärtechnik und auch Automatisierungstechnik eine wichtige Rolle, wie auf Basis der Interviews auch festzustellen ist. Dies erklärt, warum sich geeignete Interviewkandidaten in diesen Domänen befinden. Des Weiteren wurden jeweils drei Personen aus Unternehmen B und zwei Personen aus Unternehmen A befragt, sodass sich die Unternehmensdomänen der Teilnehmer zwischen den Personen im selben Unternehmen nicht unterscheiden. Abschließend ist die Interviewlänge der einzelnen Interviews zu betrachten. Hierbei wurde festgestellt, dass die vorher festgelegte Maximalzeit in fünf der acht Interviews überschritten wurde, sodass sich die durchschnittliche Interviewdauer auf insgesamt eine Stunde, 4 Minuten und 45 Sekunden beläuft. Dennoch ist eine geringe Abweichung in der Länge der Interviews im Gesamten festzustellen, da die mittlere Abweichung 19 Minuten beträgt. Die langen Interviewzeiten sind auf die zahlreichen optionalen Fragen, die in den meisten Fällen gestellt wurden, die langsame Aussprache einiger Teilnehmer sowie die ausführlichen Antworten auf die Fragen zurückzuführen.

Tabelle 5.1: Arbeitserfahrung im Bereich Softwareentwicklung

Interviewperson	**Jahre in der Softwareentwicklung**	**Jahre im aktuellen Unternehmen**
Interviewter 1	25 Jahre	3 Jahre
Interviewter 2	40 Jahre	13 Jahre
Interviewter 3	26 Jahre	8 Jahre
Interviewter 4	10 Jahre	3 Jahre
Interviewter 5	8 Jahre	4 Jahre
Interviewter 6	39 Jahre	17 Jahre
Interviewter 7	11 Jahre	4 Jahre
Interviewter 8	3 Jahre	1 Jahr

Tabelle 5.2: Rollen und Tätigkeitsbereich der Interviewpartner

Interviewperson	**Unternehmensdomäne**	**Rolle**
Interviewter 1	Echtzeitbetriebssystem	Functional Safety Manager
Interviewter 2	Echtzeitbetriebssystem	Prinzipal Softwareentwickler
Interviewter 3	Gatewayelektronik	Teamleiter Softwareentwicklung
Interviewter 4	Gatewayelektronik	Systemarchitekt
Interviewter 5	Haushaltselektronik	Testingenieur
Interviewter 6	Militärtechnik	Senior Softwareentwickler
Interviewter 7	Gatewayelektronik	Testmanager
Interviewter 8	Automatisierungstechnik	Softwareentwickler

Tabelle 5.3: Interviewpartner und ihre Unternehmen

Unternehmen	**Unternehmensdomäne**	**Interviewten**
Unternehmen A	Echtzeitbetriebssystem	I1, I2
Unternehmen B	Gatewayelektronik	I3, I4, I7
Unternehmen C	Haushaltselektronik	I5
Unternehmen D	Militärtechnik	I6
Unternehmen E	Automatisierungstechnik	I8

5.2 Ergebnisse

In den folgenden Kapiteln werden die Ergebnisse auf Basis der durchgeführten Interviews präsentiert. Dabei erfolgt eine Trennung der Ergebnisse zwischen den beiden Forschungsfragen, um eine separate Beantwortung zu ermöglichen. Um eine anschauliche Präsentation der Ergebnisse zu gewährleisten, erfolgt eine Zuordnung der jeweiligen Codes zu ähnlichen Aspekten als Code sowie zu den Interviewten.

5.2.1 Ergebnisse „Was sind sicherheitsrelevante Artefakte in der Praxis?“

Es werden zunächst die jeweiligen genannten sicherheitsrelevanten Artefakte vorgestellt und darauffolgend die Erläuterungen auf die verschiedenen Antworten, die begründen, wie die Einstufung der Sicherheitsrelevanz durchgeführt wurde. In der Tabelle 5.4 ist zu sehen, dass bei den acht Interviews 32 verschiedenste Artefakte genannt wurden. Die Anzahl der Nennungen der Artefakte in den jeweiligen Interviews unterscheidet sich hierbei von Artefakt zu Artefakt.

Tabelle 5.4: Übersicht der sicherheitsrelevanten Artefakte

Artefakte	**ähnliche Artefakte**	**Interviewte (Anzahl)**
Projektbeschreibung	-	I4 (1)
Geolokalisationsdaten	-	I6 (1)
Abbruchkriterium	-	I6 (1)
Angriffsvektoren	-	I4 (1)
Residual Risk	-	I1 (1)
Hazard Risk Mitigations	-	I1, I4 (2)
Tickets	-	I4, I5 (2)
Lasten- und Pflichtenhefte	-	I5 (1)
Sicherheitsstandards	ASIL Level; MISRA; Style Guides; Prozessentwicklungsplan; ISO 25010; ISO 26262; ISO 62443; ISO 27001; NIST; KRITIS; DSGVO; RED (Radio Equipment Directive)	I1, I2, I3, I4, I5, I6, I7, I8 (8)
Master Traceability Table	-	I1 (1)
Dokumentation	Safety Manual	I1, I2 (2)
Statische Codeanalyse Artefakte	-	I2 (1)
Design	Hardwaredesign; Softwaredesign; Systemarchitektur	I2, I5, I8 (3)
Authentifizierungskomponenten	-	I3 (1)
Verschlüsselungskomponenten	WLAN-Verschlüsselung; TFA; TFM	I3, I4, I5, I7 (4)

Artefakte	**ähnliche Artefakte**	**Interviewte (Anzahl)**
Safety Requirements	Safety Manual Requirements; Anforderungen seitens Last- und Pflichtenheft; Anforderung an das Produkt; Marktanforderung; Kundenanforderung	I1, I2, I4, I5, I6, I7, I8 (7)
Testfälle	Physischer Test; Negativtest	I1, I2, I5, I7, I8 (5)
Software Zwischenstände Hardware Zwischenstände Gerät mit Netzwerkzugang	- - -	I8 (1) I8 (1) I3 (1)
Threat- und Fehleranalyse	-	I1, I4, I6 (3)
Softwarekomponente mit Netzwerkzugang	-	I3, I7 (2)
Zertifizierungen	-	I2, I5 (2)
Reviews	-	I2, I5 (2)
Dokumentation	Safety Manual	I1, I4 (1)
Spezifikation	-	I2 (1)
Sichere Software	Gasbrennersteuerungen; Kippmomentschaltung; Rückstoßmanagement und Stabilitätsüberprüfung; Topfdeckeltemperatursteuerung; Stopp Automatik; Bremssteuerung	I2, I3, I4, I6, I8 (5)
Nutzerbediente Hardware	Notschalter; Baupläne; Layouts; Schaltzeichnungen	I5 (1)
Datenmodell	Datenmodell von Heizungsgerät	I7 (1)
Personenbezogene Daten Evil User Stories Implementierung	Passwörter; Zertifikat - Geolokalisationscode	I4, I5 (2) I4 (1) I2, I3, I4, I6 (4)

In 5.5 sind die fünf häufigsten Artefakte in der Tabelle zu sehen. Die Häufigkeiten reichen hier von vier bis acht.

Neben den Artefakten und den jeweiligen Häufigkeiten, in denen sie in den Interviews erwähnt wurden, wurden auch die jeweiligen Begründungen

Tabelle 5.5: Die häufigsten sicherheitsrelevanten Artefakte

Artefakt-Typen	**Häufigkeiten**
Sicherheitsstandards	8
Safety Requirements	7
Testfälle	5
Sichere Software	5
Implementierung	4

der Interviewpartner für die Sicherheitsrelevanz der Artefakte ermittelt. Diese sind in der Tabelle 5.6 zu sehen. Diese Aspekte wurden ermittelt, indem gefragt wurde, wie die Artefakte zur Sicherheit betragen, wie diese identifiziert werden und anhand welcher Eigenschaften dies getan wird. Ähnlich zu den Sicherheitsartefakten wurden hier in den Interviews eine Vielzahl an Gründen genannt, insgesamt 31. Es wurden Gründe genannt, die als Basis gesetzliche und normative Anforderungen, Sicherheit als funktionale und physikalische Eigenschaft, Prozess- und Entwicklungsaspekte und Analyse und Identifikation von Risiken haben. Dabei wurden domänenspezifische Konzepte erwähnt. Mixed-Criticality ist z.B. definiert als ein System, dass Komponenten beinhaltet, die eine unterschiedliche Einstufung der Sicherheitskritikalität haben [16]. Bei den Sicherheitsleveln wurde ASIL erwähnt, welches das Sicherheitsrisiko von Szenarien in der Automobildomäne ermittelt. Die häufigsten Gründe sind in Tabelle 5.7 zu sehen. Die jeweiligen Sicherheitsstandards wurden wie bei den Sicherheitsartefakten von allen Teilnehmern genannt. Diese wurden als Grund für die Sicherheitsrelevanz von Artefakten klassifiziert, da diese bei allen Interviews als ein Aspekt zur Festlegung der Sicherheitsrelevanz erwähnt wurden. Weitere Gründe sind z.B. dass gewisse Anforderung dies festlegen oder dass sicherheitsrelevante Artefakte von Safety Assesoren bestimmt werden.

Tabelle 5.6: Übersicht der Begründungen für die sicherheitsrelevanten Artefakte

Gründe	**ähnliche Gründe**	**Interviewte (Anzahl)**
Abnahme durch Software Safety Assesor	-	I1, I5, I8 (3)

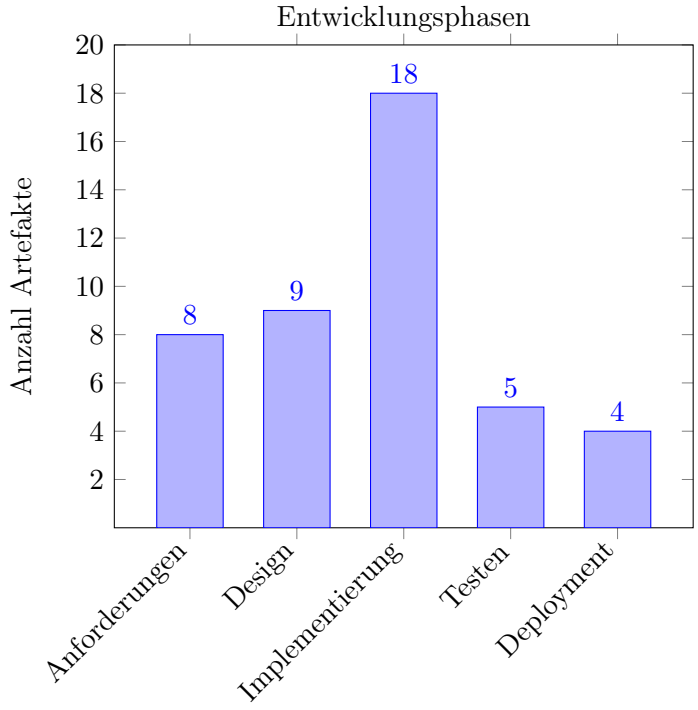
Fortgeführt auf der nächsten Seite

Gründe	**ähnliche Gründe**	**Interviewte (Anzahl)**
Anforderung setzt es voraus	Anforderung an das Produkt; Anforderung seitens Lasten- und Pflichtenheft; Anforderung seitens der Norm; Kundenanforderung; Marketinganforderung; nicht sicherheitsrelevante Anforderung	I1, I2, I5, I7, I8 (5)
Anfälligkeit für Infiltration	-	I3, I4 (2)
Prozesse	Artefakte als Erzeugnis von Prozessen; Safety Prozess legt Sicherheitsrelevanz fest; Interne Prozesse; Strengere Prozesse bei Sicherheitsrelevanz; Artefakte als Basis für andere Artefakte	I1, I2, I5 (3)
Auswirkungen auf das Gesamtsystem durch Artefakt	-	I4 (1)
Domänenspezifische Kennzeichnungen	-	I1 (1)
Erkennung durch Fehlersystem	-	I6 (1)
Erreichung eines bestimmten Sicherheitslevels	-	I1, I2, I6 (3)
Existenz der Hazard und Risk Analyse	-	I1 (1)
Finanzielle Kosten	-	I6 (1)
Gesetzliche Vorgaben	-	I5, I4 (2)
Gewährleistung von physischer Sicherheit	Isolation der Hardwarefunktionen; Redundanzen von Hardware; Schutz am Leib und Leben; Sicherheit der Hardware; Softwareseitige Hardwareisolation	I5, I6, I8 (3)
Identifikation anderer sicherheitsrelevanten Artefakte	-	I8, I4 (2)
Ermessen der Sicherheit durch Traceability	-	I6, I2 (2)
Implizite Annahme	-	I3, I8 (2)
Leitfaden für Safety	-	I1 (1)

Fortgeführt auf der nächsten Seite

Gründe	**ähnliche Gründe**	**Interviewte (Anzahl)**
Längere Verifikationszeiten	-	I6 (1)
Mixed-Criticality	Gewisser Anteil an sicherheitsrelevanten Artefakten bestimmt Sicherheit	I2 (1)
Pflicht durch Safety Standards und Richtlinien	ASIL Level; DSGVO; Hardwarenormen; IEC 62443; ISO 25010; ISO 26262; ISO 27001; KRITIS; MISRA; NIST; Prozessentwicklungsplan (PEP); RED; Style Guides; Verhinderung von Nebenläufigkeit	I7, I3, I1, I2, I4, I8, I5, I6 (8)
Rückführung auf Artefakte	Rückführung auf Dokumentation; Rückführung auf Spezifikation; Rückführung auf Zertifizierungspapier	I2 (1)
Safety Deklaration durch Tool	Safety durch Safety Artefakte	I1, I2, I5 (3)
Schutz des Unternehmens	-	I4 (1)
Sichere Software als Erzeugnis	Sicherheit als funktionale Eigenschaft; Sicherheit nach Außen; Sicherheit innerhalb eines Ökosystems; sichere Kommunikation in Komponenten innerhalb	I2, I7 (1)
Sicherer Softwareentwicklungszyklus legt Artefakte fest	-	I1 (1)
Strengere Handhabung der Artefakte	-	I2 (1)
Testing	Featuretesting; Testing der Eigenschaften des Netzwerkes; Hardwaretesting; Unterschiedliche Tests bezogen auf Sicherheitsaspekte	I5, I7, I8 (3)
Validierung der Anforderung	-	I8, I1 (2)
Verweis auf Legacy	-	I2 (1)
Entscheidung der Sicherheitsrelevanz durch Unternehmen	Teambasierte Festlegung der Sicherheit; Entscheidung durch Management	I2, I5 (2)

Abbildung 5.2: Histogramm der Artefakte in den Entwicklungsphasen



5.2.2 Ergebnisse „Wird Tracing für diese Artefakte betrieben und wenn ja, wie?“

Die Beantwortung der zweiten Forschungsfrage erfolgte durch Befragung der Interviewpartner in drei unterschiedlichen Fragenblöcken zum Tracing von sicherheitsrelevanten Artefakten. Zunächst wurde eruiert, ob die Interviewpartner in ihren Unternehmen Tracing für sicherheitsrelevante Artefakte betreiben. Hierbei wurde deutlich, dass alle Interviewpartner der Auffassung sind, dass Tracing unabhängig davon, ob die Artefakte sicherheitsrelevant oder nicht-sicherheitsrelevant sind, eine aktive Rolle spielt. Die Interviewten 3 und 4 gaben an, dass Tracing nicht in großem Umfang umgesetzt wird, jedoch aktiv versucht wird, die Traceability zwischen Artefakten auszubauen. Bei der Frage, ob das Tracing von sicherheitsrelevanten Artefakten gesondert behandelt wird, divergieren die Ergebnisse von denen der vorherigen Fragen. Eine Übersicht findet sich in Tabelle 5.8. Vier von acht interviewten Personen (Interviewte 1, 2, 5, 6) gaben an, dass die Behandlung von Tracing für sicherheitsrelevante Artefakte im Gegensatz zu generischen Artefakten unterschiedlich erfolgt. In Bezug auf die Behandlung von Tracing für sicherheitsrelevante Artefakte im Vergleich zu generischen Artefakten hat Interviewter 1 dargelegt, dass die Notwendigkeit für Traceability durch die Sicherheitsstandards vorgegeben wird. Dies führt dazu, dass Traceability

Tabelle 5.7: Meistgenannte Gründe und ihre Häufigkeiten

Grund	**Häufigkeit**
Pflicht durch Safety Standards und Richtlinien	8
Anforderung setzt es voraus	5
Testing	3
Gewährleistung von physischer Sicherheit	3
Erreichung eines bestimmten Sicherheitslevels	3
Abnahme durch Software Safety Assessor	3
Prozesse	3

für generische Artefakte für das Team mit nicht effektivem Mehraufwand verbunden ist. Weitere Gründe und Unterschiede sind in Tabelle 5.9 zu sehen. Interviewpartner 6 hat domänenspezifische Unterschiede in der Militärtechnik beim Tracing von sicherheitsrelevanten Artefakten genannt, die in den anderen Domänen nicht vorzufinden sind. Die anderen Interviewpartner haben auf strengere gesetzliche Bestimmungen bei Traceability und spezialisierte Tools beim Tracing von sicherheitsrelevanten Artefakten verwiesen. Die restlichen vier Teilnehmer haben in der Praxis keine Unterschiede beim Tracing zwischen den generischen und sicherheitsrelevanten Artefakten festgestellt. Der Interviewte 4 gab an, dass das Tracing von sicherheitsrelevanten Artefakten in Planung sei, da es in Zukunft aus Seiten des Gesetzes gefordert sein wird. Die übrigen Teilnehmer nannten diverse andere Gründe, warum keine Differenzierung des Tracings in Bezug zur Sicherheitsrelevanz besteht. Diese werden in Tabelle 5.10 aufgelistet. Als weitere Gründe wurden genannt, dass das Tracing allgemein gehalten ist und somit keine Unterscheidung besteht, sowie dass Sicherheitsaspekte nicht explizit getrennt als nicht funktionale Eigenschaft behandelt werden.

Tabelle 5.8: Verteilung der Unterscheidung zwischen nicht-sicherheitsrelevanten und sicherheitsrelevanten Artefakten

Unterschied vorhanden	**Interviewpartner**
Tracing wird nur für sicherheitsrelevante Artefakte betrieben	I1
Unterschiede vorhanden	I2, I5, I6
Unterschiede nicht vorhanden	I3, I4, I7, I8

Tabelle 5.9: Unterschiede von Tracing von sicherheitsrelevantem Artefakte zu nicht-sicherheitsrelevanten Artefakten (Gruppe Unterschied zwischen den Artefakten)

Unterschiede	**Interviewpartner**
Beweis der sicherheitsrelevanten Traceability	I5
Gesetzliche Bestimmungen	I5
Mehr und spezialisierte Tools bei sicherheitsrelevanten Artefakten	I2
Notwendig vom Safety Standard	I1
Persistente Fehlermeldungen	I6
Remote an- und ausschaltbar	I6
Safety Assessor überprüft Traceability	I1
Synchronisation der Tracingzeiten	I6
Traces werden in einer Datenbank gespeichert für Nachweise	I6
Tracing wird gemacht, aber durch externe Tools	I6
Verschiedene Stufen an Tracing	I6

Tabelle 5.10: Gründe, warum es keine Unterschiede zwischen dem Tracing gibt (Gruppe kein Unterschied zwischen den Artefakten)

Gründe	**ähnliche Gründe**	**Interviewpartner**
Fehlende Konsistenz des Tracings	-	I3
Fehlende Zeit	-	I3
Fehlendes Sicherheitsbewusstsein	-	I4
Hohe finanzielle Kosten	-	I4
In Planung sicherheitsrelevantes Tracing umzusetzen, da in Zukunft gezwungen	Noch nicht gesetzlich verpflichtend	I4
Sicherheitsrelevante Aspekte werden aus funktionaler Sicht betrachtet	-	I7
Tracing als grundlegender Prozess deswegen keine Unterscheidung	-	I8

Fortgeführt auf der nächsten Seite

Gründe	ähnliche Gründe	Interviewpartner
Tracing für generische Artefakte	Fokus auf Requirements und System Software Architektur; Priorität auf Artefakte mit höherem Stellenwert	I3
Tracing generell beim Endkunden, aber nicht für interne Artefakte	-	I4
Tracing primär für die Zertifizierung	-	I8
Mehrheit der Artefakte sicherheitskritisch, womit kein Bedarf der Differenzierung besteht	-	I8

Neben den Gründen für die Unterscheidung (oder fehlende Unterscheidung) beim Tracing wurden Beispiele für bestehende Trace-Links von sicherheitsrelevanten Artefakten gefragt und Vorteile der Traceability. Die genannten Trace-Links sind in Tabelle 5.11 zu sehen. Die Pfeile drücken aus, dass ein Trace Link zwischen zwei Artefakten besteht. Alle Teilnehmer haben jeweils ein Trace-Link nennen können, der Interviewte 4 hat jeweils drei Trace-Links genannt. Die Trace-Links sind im Ganzen alle unterschiedlich zueinander, Anforderungen und Testfälle wurden aber von allen Interviewpartnern außer Interviewpartner 6 genannt, da dieser Artefakte genannt, die primär in der Militärtechnik eine Rolle spielen.

Tabelle 5.11: Übersicht der Trace-Links

Trace-Links	Interviewpartner
Hazard -> Risk Mitigation -> Software Requirement -> Test Case	I1
JAMA-Projekt -> Jira-Issue -> Implementierung -> Gitlab Merge Request (Review) -> Testing -> Build Job -> Sichere Software	I2
Requirements -> Systemarchitektur umd Systemrequirements -> Systemtests	I3
Angriffsvektoren -> Software- oder Systemkomponenten -> Mitigation -> Anforderungen -> Tickets	I4
Sicherheitsdaten -> Software- oder Systemarchitektur -> Software- oder Systemkomponenten	I4
Tickets -> Commits Git Repository	I4

Fortgeführt auf der nächsten Seite

Trace-Links	**Interviewpartner**
Tickets -> Hauptanforderung -> Anforderung -> Entwicklungsarbeit -> Testfall -> Testausführung	I5
Ist-Position -> Zielposition -> Regelung -> Abbruchkriterium -> Wartung	I6
Anforderung -> Test -> Software	I7
Globale Anforderung -> Kleine Anforderungen -> Test -> Soft- und Hardwarestand	I8

Die Vorteile sind in Tabelle 5.12 zu sehen. Sechs von acht Teilnehmer haben unterschiedlichste Vorteile genannt, die sich unter der Nachvollziehbarkeit der Artefaktbeziehungen zusammenfassen lassen. Ein Vorteil war es zudem, dass durch das Auffinden der Fehler durch die Traceability der Prozess insgesamt verbessert werden kann. Zusätzlich erlaubt Traceability die Versionierung von ganzen Entwicklungsständen. Dies heißt, dass Traceability es erlaubt Entwicklungsstände zu protokollieren, mit den zugehörigen Trace Points und den Trace Links.

Tabelle 5.12: Vorteile des Tracings

Vorteile	**ähnliche Vorteile**	**Interviewpartner**
Abdeckung der Tests bezogen auf Requirements	-	I3
Nachvollziehbarkeit der Artefaktbeziehungen	Fehleridentifikation; Sicherheitslücken nachvollziehen; Softwareversion nachverfolgen; Zertifizierung durch Security Assessor nachvollziehen	I1, I2, I4, I6, I7, I8
Prozessverbesserung	-	I2
Rechtliches Absichern bei Schäden	Nachweisen der Non-Regression der Fehler (keine Rückwirkungen von Änderungen)	I6
Regeln des Gesetzgebers werden erfüllt	Eigene Daten werden geschützt	I4
Sicherheit, dass das Große Ganze fertig ist	-	I5
Versionierung durch Traceability	-	I6
Informationen sind kondensiert	-	I8

Tabelle 5.12: Vorteile des Tracings

Vorteile	ähnliche Vorteile	Interviewpartner
Dokumentation der Entwicklungsinformationen	Wissen geht nicht verloren durch Mitarbeiterfluktuation	I7

5.2.3 Mit welchen Methoden und Prozessen wird dieses Tracing durchgeführt?

Im weiteren Fragenblock wurden Methoden und Prozesse der Traceability von sicherheitsrelevanten Artefakten erfragt. Die ersten beiden Fragen bezogen sich auf die für das Tracing eingesetzten Tools sowie deren Zweck und Kontext. In Tabelle 5.13 werden die eingesetzten Tools und die genannten Gründe für die sicherheitsrelevanten Artefakte aufgelistet. Die Mehrheit der Befragten gab an, die Tools auch für nicht sicherheitsrelevante Artefakte zu nutzen. Explizit für sicherheitsrelevante Artefakte wurden JAMA, itemis Secure und Wireshark genannt. In nahezu allen Interviews (7 von 8 Interviews) wurde Jira genannt, primär zum Verwalten der Aufgaben, die im Entwicklungsprozess auftreten. Weitere Tools, die in den Interviews genannt wurden, sind Polarion, JAMA und Enterprise Architect. Diese sind ebenfalls in Tabelle 5.14 aufgeführt.

Tabelle 5.13: Tools für die sicherheitsrelevanten Artefakte

Tools	ähnliche Tools	Interviewpartner
IBM DOORS	Scheitert nicht und ist umfangreich; Traceability von Anforderung bis Software	I6
Google Drive	Archivierung von Dokumenten	I5
itemis Analyze	Schnittstelle zwischen verschiedenen Traceability Tools; Traceability zwischen Artefakten darstellen und verwalten	I4
itemis Secure	Sicherheitsrisiken erfassen; TARA Analyse	I4
Wireshark	Verschlüsselungsmethoden können im Netzwerk identifiziert werden; Netzwerkverkehr zwischen Komponenten können untersucht werden	I7
Confluence	Dokumentation	I7
Jira	Verwalten von Tickets; Verlinkung zwischen Anforderung und Aufgaben, Verlinkung zwischen verschiedenen Jira Tickets; Tracing der Arbeit	I1, I2, I3, I4, I5, I7, I8

Fortgeführt auf der nächsten Seite

Tools	**ähnliche Tools**	**Interviewpartner**
Enterprise Architect	Software Architektur; Modellieren mit UML und SysML; Verwaltung von sicherheitsbezogenen Anforderungen; Threat Modellierung zur Erfassung der risikobehafteten Komponenten; Risikoanalyse	I3, I4
Robot Framework	Aufsetzen und Organisieren von Tests	I3
Polarion	Anforderungsmanagement; Gesamter Entwicklungszyklus mit Polarion verwaltbar; Testmanagement als Zusatzfeature	I3, I4, I7, I8
Gitlab	Automatische Tests; Traceability zwischen Jira und Source Code	I2
Python Skripte	Dateigenerierung von Artefakten	I1
JAMA	Anforderungsmanagement; Artefakte extrahiert aus REST API; Große Cloud Datenbank	I1, I2, I4

Tabelle 5.14: Die häufigsten Toools für das Tracing von sicherheitsrelevanten Artefakten

Tools	**Häufigkeiten**
Jira	7
Polarion	4
JAMA	3
Enterprise Architect	2

Es wurden zudem Fragen zu Compliance und Audits gestellt. Die Antworten sind in 5.15 zu sehen. Die Einhaltung von Standards und Normen im Tracing-Prozess ist entscheidend und erfordert detaillierte Dokumentation, kontinuierliche Risikobewertungen sowie Nachweisbarkeit durch Pentests und TÜV-Prüfsiegel, um Produktanforderungen zu erfüllen und die Marktfreigabe zu gewährleisten. Audits im Tracing-Prozess werden iterativ und unabhängig durchgeführt, oft durch externe Firmen und Prüfinstanzen, um die Nachvollziehbarkeit des Entwicklungsprozesses zu beurteilen und die Einhaltung der Normen zu verifizieren, wobei regelmäßig Zertifizierungen und Prüfungen von sicherheitsrelevanten Artefakten stattfinden.

Tabelle 5.15: Compliance und Audits

Compliance und Audit Aspekte	**ähnliche Compliance und Audit Aspekte**	**Interviewpartner**
Spielt eine große Rolle	-	I1
Sicherheitsassessor für die Beurteilung	Überprüft ob die Traceability Regeln eingehalten werden; Schwachstellen in dem Prozess aufdecken	I1, I2, I7
Internes Auditing um die Traceability Qualität zu überprüfen	-	I1
Risikobetrachtungen müssen gemacht werden	HARA; Hazard und Risk Analyse	I2
Garantie, dass dem Prozess gefolgt wurde	Produktanforderungen sind erfüllt	I2
Keine Beispiele für Compliance Herausforderungen	Keine praktische Erfahrung; Keine Probleme mit IT-sicherheitsrelevanter Software	I2, I3
Keine Beispiele für Audits, da Tracing anderen Fokus hat	-	I3
Nachweisbarkeit oder Dokumentation gefordert	RED ab August 2025 gefordert; Nachweisen durch Pentests	I4
Prüfsiegel durch TÜV Audit	Traceability im Entwicklungsprozess überprüfen; Prüfsiegel essenziell um das Produkt auf den Markt zu bringen	I4, I5
Artefakte für Zertifizierungen werden mit separaten Tools verwaltet	Archivierung hat eine höhere Relevanz	I5
Iterativer Prozess im Audit	Designdokumente werden wiederholend auf Eigenschaften überprüft	I5
Prüfinstanz beurteilt Nachvollziehbarkeit des Entwicklungsprozesses	-	I6

Fortgeführt auf der nächsten Seite

Compliance und Audit Aspekte	**ähnliche Compliance und Audit Aspekte**	**Interviewpartner**
Externe Firma beurteilt anhand FMEA Untersuchung	Tracekette für sicherheitsrelevante Objekte werden untersucht; nicht nur Source Code wird überprüft, sondern alle relevanten Artefakte wie Tests, Aufzeichnungen etc.	I6
Audits unabhängig von dem Unternehmen	Unternehmen ist zu klein für Audits; Deutsche Regierung sucht sich externen Konzern aus; Audits werden initiiert durch externen Konzern; Audit im Anschluss einer SAT; Keine Fehler in den Audits bisher gefunden	I6
Rollenverteilung bei Compliance	Requirements Ingenieure geben Anforderungen vor; Tester verifizieren die Compliance Anforderungen	I7
Spielt eine große Rolle	-	I7
Einhaltung von Standards sehr wichtig	-	I8
Am Anfang der Entwicklung ohne Auditing	Entwickler überprüfen die Normeneinhaltung selbst	I8
Entwickler haben ein Bewusstsein über die Normen	Normenwissen werden durch Lehrgänge erlangt	I8
Externe Prüfer beobachten die Einhaltung der Normen in der späteren Entwicklung	Graduell mehr werdend durch vollständige Verifizierung	I8

Im dritten Fragenblock wurden Fragen zur Prozessintegration des Tracings von sicherheitsrelevanten Artefakten gestellt. Die Fragen thematisieren die Integration des Tracing in bestehende Entwicklungs- und Qualitätssicherungsprozesse und optional diese Prozesse mit Sicherheitsaspekten, die Vorteile in Bezug auf Sicherheit und die Herausforderungen diesbezüglich. Die Ergebnisse sind in 5.16 zu sehen. Das Tracing von sicherheitsrelevanten Artefakten wird durch Jira Issue -> Commit Links, automatisierte Unittests bei Artefaktänderungen, und QMS Anweisungen, die den Tool-Einsatz und Safety Lifecycle definieren, in die bestehenden Prozesse integriert. Vorteile

bietet das Tracing, da Sicherheitslücken im gesamten Prozess betrachtet werden, was zu einem sicheren Produkt durch die Vollständigkeit und Nachvollziehbarkeit des Entwicklungsprozesses führt und hohe Kosten durch Rückrufe verhindert. Die Integration des Tracings erfordert spezialisierte Sicherheitsabteilungen, ist abhängig vom Unternehmen, und gestaltet sich aufgrund der Schwierigkeit der Verbindung mehrerer Tools und der Notwendigkeit einer aktuellen Traceability herausfordernd.

Tabelle 5.16: Integration in die Entwicklungsprozesse

Prozessaspekte	**ähnliche Prozessaspekte**	**Interviewpartner**
Bei größeren Änderungen an dem Softwarestand werden Probedurchläufe durchgeführt	-	I6
Bewusstsein für Security notwendig	Auswirkungen fehlender Sicherheit können nicht eingeschätzt werden; Mit hohen Kosten verbunden; Mitarbeiter müssen in Security geschult sein; Eigene Abteilung für Security notwendig damit Sicherheitsaspekte eingehalten werden	I4, I7
Überprüfung und Analyse	Check auf aktuelle Sicherheitslücken; Check der Code Reviews; Statische Codeanalyse; MISRA mit Coverity; Strenge Regelsätze bei sicherheitsrelevanten Artefakten bei MISRA	I2, I3
Diverse Methodiken für Einbindung in Prozess	-	I4
Features aus V-Modell ziehen	-	I3
Ganzheitlicher Traceability Prozess verhindert hohe Kosten durch Rückruf	-	I5

Fortgeführt auf der nächsten Seite

Prozessaspekte	**ähnliche Prozessaspekte**	**Interviewpartner**
Guter Entwicklungsprozess nötig vor Fokus auf Sicherheit in der Entwicklung	-	I4
Herausforderungen und Anpassungen	Prozessdurchführung kostet Zeit und Geld	I1, I7
Jira Issue -> Commit Link technisch erzwungen	Verweis auf JAMA Projekt damit möglich für sicherheitsrelevante Artefakte	I2
Kein dedizierter Sicherheitszyklus	Auf Artefaktebene gleichwertig betrachtet; Entwicklung muss sicher sein als einzige Leitlinie; Sicherheitskomponenten sind ein kleiner Bestandteil	I5, I8
Keine speziellen Integrationsschritte	Die selben Tools für die Traceability von allen Artefakten	I5, I7
Mehrere Tools sind schwierig miteinander zu verbinden	-	I4
Paralleles Hardware Tracing bei zeitkritischen Artefakten	Separater Check zur Überprüfung der Validität vor Start der Rakete; Unit wird im Labor Szenarios in der Realität vorgespielt	I6
Prozessbestandteile mit Fokus auf Softwarequalität nicht Sicherheit	-	I3
QMS Anweisungen	Definieren den Tool-Einsatz; Definieren die Artefakte; Definieren Safety Lifecycle	I1
Qualitätssicherung passt sich Traceability an	Automatischer Unit Test bei Änderung von bestimmten Artefakten; Ergebnisse der Qualitätssicherung kommen in Polarion; Qualitätsmanagement bearbeitet diese im Nachgang	I6, I8
Risk Analyse eher passend für End-to-End Domäne	-	I4

Fortgeführt auf der nächsten Seite

Prozessaspekte	**ähnliche Prozessaspekte**	**Interviewpartner**
Safety Assurance durch Traceability	Sicherheitslücken im Kontext des gesamten Prozesses betrachten	I1, I4, I8
Schlechte Erfahrungswerte in der Umsetzung von Traceability mit Einbeziehung von Security	-	I4
Schwierigkeit der Implementierung der Traceability abhängig vom Unternehmen	Automobilunternehmen implementieren Traceability gut; Große Konzerne teilweise gute Implementierung von Traceability	I4
Sicheres Produkt durch Vollständigkeit und Nachvollziehbarkeit des Prozesses	-	I2, I7
Softwarekonstrukt der Traceability behindert Entwicklungserfolg	-	I3
Strikte Regeln im Entwicklungsprozess	Jede Anforderung muss getestet werden	I5
Traceability aktuell halten schwierig	-	I3
Traceability als zentraler Teil des Produktlebenszyklus	Andere Prozessteile passen sich an das Traceability an; Keine Anpassung an der Traceability selbst notwendig	I8
Traceability spielt bei der Versionierung der Systeme eine Rolle	Prüfabläufe in der Versionierung werden genutzt um bei Aufkommen von Fehler Probleme zu beheben	I6
Überprüfung der Sicherheit primär seitens des Managements	Management hat mehr Überblick über die Forderungen des Safety Assesors	I5

Die Interviewteilnehmer wurden zum Stand der Automatisierung des Tracings innerhalb des Unternehmens gefragt. Die Interviewten 3 und 4 haben angegeben, dass keine Automatisierung besteht. Gründe sind hier z.B. dass Komponenten des Tracingsprozess nicht automatisiert werden können oder dass manuelle Tracingtools bevorzugt werden, da sie komfortabel zu nutzen sind, welche in Tabelle 5.17 zu sehen ist.

Die restlichen sechs Teilnehmer haben ausgesagt, dass der Tracingprozess

Tabelle 5.17: Gründe für die fehlende Automatisierung im Tracingprozess

Gründe	**Ähnliche Gründe**	**Interviewpartner**
Bestimme Komponenten des Tracings kann man nicht automatisieren	Analyse der Security Lücken oder Assets geschieht manuell	I4
Code-Analyse-Tools haben hohen Aufwand, um false positives zu verhindern	-	I4
Kein ordentlicher Entwicklungsprozess im Allgemeinen, weswegen Automatisierung und Security noch keine Priorität hat	-	I4
Manuell aus Gründen der Tools, Komfort und Usability	Kein Tracingtool gefunden, wo alle Entwicklungsabteile zufriedengestellt worden sind	I3
Requirements werden manuell miteingetragen	-	I3

semi-automatisiert ist, also teils manuelle und teils automatisierte Komponenten hat. Manuelle und automatisierte Komponente und generellen Feststellungen über die Automatisierung des Tracing Prozesses wurden in Tabelle 5.18 aufgelistet.

Tabelle 5.18: Manuelle und automatisierte Komponenten im semi-automatisierten Prozess

Aspekte zur Semi-Automatisierung	**ähnliche Aspekte zur Semi-Automatisierung**	**Interviewpartner**
Automatisierung der Artefaktlokalisiation und Generierung einer Master Traceability Table	Manuelle Korrektur der Master Traceability Table notwendigv	I1
Die zeitlichen Ressourcen fehlen für die weitere Automatisierung	-	I1
Keine Sicherheit, dass Fehler bei der Automatisierung auftreten, da Safety Projekte sicherheitskritisch sind	-	I1

Fortgeführt auf der nächsten Seite

Aspekte zur Semi-Automatisierung	**ähnliche Aspekte zur Semi-Automatisierung**	**Interviewpartner**
Das jeweilige Jira Ticket muss manuell angegeben werden	-	I2
Integration der Implementierung ist automatisiert	Buildsystem ist automatisiert	I2
Die Übertragung der Artefakte zwischen Tools gestaltet sich schwierig	-	I2
Validierung der Artefakte muss manuell erfolgen	-	I2
Automatisierung von Vorschlägen für Trace Links wird als problematisch angesehen	Automatisierung der Schritte in der Entwurfsphase bis zur Implementierung gestalten sich schwierig	I2
Die Abdeckung der Tests bezogen auf die Anforderung ist automatisiert	Ausgabe der Automatisierung ist das Fehlen der Tests oder die Vollständigkeit der Traceability	I5
Von Requirements bis zur Spezifikation wird automatisiert	Bis zur Implementierungsebene besteht keine Automatisierung	I6
Statische Code Analyse Tools werden verwendet	Understand C++, MISRA, Code Analyse Tools liefern Ergebnisse, die nicht realitätsnah sind	I6
Einige Code Artefakte können nicht automatisiert werden, da kein Tracing möglich ist	Code der dynamischen Systemzustände unterworfen ist	I6
Testautomatisierung Tools tragen zur Automatisierung bei	Test Skripten sind in einer Tracingkette mit Test-Ticket und Anforderungen	I7
Einige ältere Geräte haben Funktionalitäten, die nicht mittels Automatisierung geprüft werden können	-	I7
Großer Anteil an manuellen Unittests	Im fertigen Produktsystem konnte man gut automatisieren, Testergebnisse waren dadurch weniger möglich getraced zu werden	I8

Fortgeführt auf der nächsten Seite

Aspekte zur Semi-Automatisierung	ähnliche Aspekte zur Semi-Automatisierung	Interviewpartner
Die zeitlichen Ressourcen fehlen für die weitere Automatisierung	Erfahrene Tester für die Automatisierung haben gefehlt; Tracing aufbauend auf Legacy Methodiken, weswegen Automatisierung sich schwieriger gestalten	I8

Zusätzlich zu der Art der Automatisierung wurden Vorteile und Nachteile der Automatisierung im Tracing Prozess erfragt. Die von nahezu allen Teilnehmern genannten Vorteile umfassen die Reduktion der manuellen Tätigkeiten im Tracing sowie die Verringerung der Fehleranfälligkeit. Eine Übersicht der genannten Vorteile ist in Tabelle 5.19 zu finden. Die genannten Nachteile der Automatisierung wurden von den Teilnehmern 2, 3 und 7 verneint. Als Nachteile, die von den übrigen Interviewten genannt wurden, sind beispielsweise die Fehleranfälligkeit aktuell bestehender Automatisierungen sowie die Abhängigkeit von extern geschriebenen Tools zu nennen. Eine Übersicht aller genannten Nachteile findet sich in Tabelle 5.20.

Tabelle 5.19: Vorteile der Tracingautomatisierung

Vorteile	ähnliche Vorteile	Interviewpartner
Zusammenbringen der geleisteten Arbeit der Mitglieder	-	I8
Man erweitert seinen Kenntnisstand	-	I7
Verfolgbarkeit durch Automatisierung der Traceability	-	I7
Schnellere Ergebnisse	-	I5, I7
Keine Vorteile, da es keine Automatisierung gibt	-	I3
Fehleranfälligkeit durch manuelle Arbeit wird reduziert	Fehler durch niedrige Kenntnisstände einzelner unerfahrener Mitarbeiter werden ausgeglichen; Automatisierung gibt Sicherheit, dass es keine Fehler gibt	I2, I4, I6, I8
Reduzierung der manuellen Arbeit für das Tracing	-	I1, I2, I4, I5, I6, I7, I8

Tabelle 5.20: Nachteile der Tracing-Automatisierung

Nachteile	**Subnachteile**	**Interviewpartner**
Aufsetzen und Instandhalten der Automatisierung kostet Zeit	-	I8
Automatisierung schafft es noch nicht feingranular auf Artefaktebene zu arbeiten	-	I5
Abhängigkeit von externen Tools	Tool kann zeitweise ausfallen und nicht verfügbar sein; Abhängigkeit auf potenziell unsichere externe Tools	I1, I5
Durch Automatisierung können Fehler entstehen	Keine Korrektur der durch Automatisierung entstandenen Fehler durch Vertrauen auf Automatisierung	I4, I5
Keine Nachteile	-	I2, I3, I7

Als fünfter und letzter Aspekt wurde abgefragt, wie soziale Aspekte in der Traceability von sicherheitsrelevanten Artefakten gefördert werden. Dazu wurde zunächst eruiert, wie die Kommunikation und Zusammenarbeit zwischen den Beteiligten im Tracingprozess gefördert wird. In diesem Kontext wurden Best Practices für die Kommunikation erfragt, sofern keine Beispiele genannt wurden. Zudem wurde eruiert, wie die rollenspezifischen Unterschiede im Tracing aussehen. Die Antworten zu diesen Fragen sind in Tabelle 5.21 zu finden. Die Abhaltung von Teammeetings in verschiedenen Ausprägungen, die Unterstützung der Kommunikation durch Einbindung verschiedener Teams sowie die Förderung der Kommunikation durch das Tracing, indem Artefakte von mehreren Teams bearbeitet werden, wurden jeweils mehrmals genannt. Die letzte Frage war, ob rollenspezifische Unterschiede zwischen dem Tracing von sicherheitsrelevanten Artefakten existieren. Die Antworten hierzu befinden sich in Tabelle 5.22. Die Unterschiede manifestieren sich beispielsweise in der Tatsache, dass verschiedene Rollen unterschiedlichen Zugriff auf Artefakte haben, an unterschiedlichen Artefakten arbeiten und auch, dass die Rollen an unterschiedlichen Phasen des Entwicklungsprozesses arbeiten. Der Interviewte 6 konnte keine Aussage treffen, da ihm kein Einblick in die Arbeit der externen Mitarbeiter gewährt wurde.

Tabelle 5.21: Soziale Aspekte in der Traceability von sicherheitsrelevanten Artefakten

sozialer Aspekt	**ähnliche soziale Aspekte**	**Interviewpartner**
Abhaltung von Teammeetings	Aufteilung in Arbeitspakete bei Folge Meetings; Kick-Off Veranstaltungen bei neuen Features, um diese Features vorzustellen; Meetings unter Kollegen zur Bestimmung der Sicherheitsrelevanz; Teammeeting in Form von monatlichen Themenbesprechungen; Vorschläge werden geäußert	I3, I5, I7
Änderung an einem komplexen System müssen durch eine offizielle Instanz abgesegnet werden	-	I6
Kommunikation wird über ein TGP Verfahren abgesichert	Sensible Informationen müssen abgesichert werden	I6
Korrekte Kommunikation im Tracing Prozess erfordert Automatisierung oder sehr viel Disziplin	Automatisierung hilft durch digitale Erinnerung im Projektmanagementprozess, dass das Tracing durchgeführt wird	I8
Kommunikation ist wichtig, um Konzepte der Traceability zu vermitteln	Kürzere Kommunikationswege sind vorteilhaft für das Tracing	I4, I8
Paired Programming wird durchgeführt	-	I7
Reviews, die an das Tracing gekoppelt sind, erlauben Validation des Tracing durch Zusammenarbeit	-	I1, I7
Software-Architekten als Schlüsselfigur für die Traceability	-	I3, I4
Teilnahme an Schulungen zum Erwerb von Kenntnissen der Traceability	-	I5

Fortgeführt auf der nächsten Seite

sozialer Aspekt	ähnliche soziale Aspekte	Interviewpartner
Tools unterstützen Kollaboration durch Einbindung von verschiedenen Teams	Können fehlende Kenntnisse zwischen Fachbereichen durch strikte Rollentrennung ausgleichen	I2, I3, I8
Tracing fördert Kommunikation, da das Tracing durch die Artefakte verschiedener Teams geht	-	I1, I2, I5

Tabelle 5.22: Rollenspezifische Unterschiede in der Traceability von sicherheitsrelevanten Artefakten

rollenspezifischer Unterschied	ähnlicher rollenspezifischer Unterschied	Interviewpartner
Verschiedene Rollen können verschiedene Anweisungen approven	Functional Safety Manager muss Test und Projektmanagement approven	I1
Verschiedene Rollen arbeiten an verschiedenen Stellen des Entwicklungsprozesses	-	I2, I4
Softwarearchitekten halten Austausch mit anderen Kollegen und halten die Verlinkung der Artefakte aktuell	-	I3
Primär beschäftigen sich Systemarchitekten und Rollen mit hohem Anforderungsmanagement mit Traceability	Rollen die keine Systemarchitekten sind beschäftigen sich weniger an der Traceability; Softwarearchitekten halten Austausch mit anderen Kollegen und halten die Verlinkung der Artefakte aktuell	I3, I4
Unterschiedliche Artefakte für unterschiedliche Rollen	Unterschiedliche Trace Links zwischen Artefakten	I5
Testmanager überprüft die Traceability der Tests	Sicherheitsrelevanz der Anforderungen wird durch Testmanager bestimmt; Tester haben die Freiheit mehr Tests zu machen	I7

Fortgeführt auf der nächsten Seite

rollenspezifischer Unterschied	ähnlicher rollenspezifischer Unterschied	Interviewpartner
Tester stellen Bugreports rein	-	I8
Projektmanager trifft Entscheidungen und verteilt Aufgaben	-	I8

5.2.4 Welche Probleme bestehen bei diesem Tracing?

Im dritten und letzten größeren Subfragestellung zu der dritten Forschungsfrage ging es um die Probleme, Nachteile und entsprechende Lösungsmaßnahmen im Tracing von sicherheitsrelevanten Artefakten. Es wurde gefragt, welche Nachteile beim Tracing bestehen, welche Probleme bei der Implementierung des Tracings bestehen, welche Sicherheitsrisiken vorzufinden sind und wie mit all diesen Problemen umgegangen wird. Die Tabelle 5.23 fasst hierbei die Probleme und Nachteile zusammen. 5 von 8 Interviewpartner haben hierbei den erhöhten Aufwand durch Traceability als Nachteil genannt. Weitere Nachteile sind z.B. dass Trace Links öfters nicht aktuell oder korrekt gehalten werden oder dass die Integration verschiedener Tracing Tools sich schwierig gestaltet. Die praktische Umsetzung von Traceability in der Softwareentwicklung ist durch die Komplexität der Integration, die Notwendigkeit spezifischer Fachkenntnisse und die Sicherstellung der Datenaktualität und -sicherheit erheblich erschwert. Die genannten Lösungsansätze sind in Tabelle 5.24 vorzufinden. Mehr als einmal genannt wurden hierbei, dass das Tracing klar definiert wird mittels Prozessen und Standards und dass das Tracing nicht als zusätzlicher Arbeitsschritt gesehen wird, sondern als integraler Bestandteil des Entwicklungsprozesses.

Tabelle 5.23: Nachteile und Probleme im Tracing von sicherheitsrelevanten Artefakten

Nachteile und Probleme	ähnliche Nachteile und Probleme	Interviewpartner
Agiles Entwicklungsmodell führt dazu, dass man Software bei Änderungen der Features neu schreiben muss	-	I5

Fortgeführt auf der nächsten Seite

Nachteile und Probleme	**ähnliche Nachteile und Probleme**	**Interviewpartner**
Aktualität der Trace Links ist nicht sichergestellt	Nicht sicher welche Version des Entwicklungsstandes aktuell; Trace Links werden teilweise gar nicht eingetragen; Trace Links nach einer langen Zeit zu verwalten gestaltet sich schwierig aufgrund fehlendem Wissen; Sicherheitsproblem durch fehlende Änderungen der Trace Links	I1, I3
Artefakte, die Informationen über Sicherheitslücken beinhalten, können böswillig ausgenutzt werden	-	I4
Automatische Tools müssen mit Berücksichtigung der Sicherheit korrekt konfiguriert werden	-	I4
Den Überblick über das Gesamte zu halten gestaltet sich komplex	-	I5
Stakeholder und Externe Kunden müssen von der Sinnhaftigkeit der Traceability überzeugt werden	Es muss beim Kunden ein Bewusstsein für Sicherheit geschafft werden; Tracing von externen Unternehmen erfordert Umdenken; Kunde muss mit höheren Kosten, erhöhten Aufwand und Veränderung der Arbeitskultur rechnen; verschiedene Entwicklungsabteilungen müssen überzeugt werden	I1, I4, I7
Gesetzliche Bestimmungen bezogen auf Sicherheit sind eine Herausforderung	-	I5
Infrastruktur der Tracing-Software muss konstant lauffähig gehalten werden	-	I7

Fortgeführt auf der nächsten Seite

Nachteile und Probleme	**ähnliche Nachteile und Probleme**	**Interviewpartner**
Integration zwischen verschiedenen Tools gestaltet sich schwierig	-	I2
Korrekte Kommunikation innerhalb des Tracings erfordert Können	-	I8
Korrektes manuelles Aufsetzen des Tracings schwierig	Erfordert Fachwissen über Tools; Manuelle Arbeit beim Tracing führt zu Fehleranfälligkeit	I1, I2
Sehr viele Schritte bei der Durchführung des Tracings	-	I1, I4
Tracing Daten werden aufgrund schwammiger Gründe lange gespeichert	-	I6
Tracing Informationen korrekt und ausführlich in Verbindung zu bringen gestaltet sich problematisch	-	I8
Tracing wird seitens Externer aus nicht genannten Gründen verhindert	-	I6
Viel Ressourcenaufwand	Finanzielle Kosten; Kann durch die langfristigen Zeiteinsparungen ausgeglichen werden; Zeit wird nur für safety-relevante Artefakte aufgewendet	I1, I4, I5, I6, I8

Tabelle 5.24: Lösungen wie mit den Problemen im Tracing umgegangen wird

Lösungsansätze	**ähnliche Lösungsansätze**	**Interviewpartner**
Es wird an generellen Entwicklungsprozessen gearbeitet	Bei der Planung müssen Externe aufgrund der anfallenden Overhead durch Tracing berücksichtigt werden	I4
Klar definieren, wie man das Tracing durchführen will	Immer die Trace Links setzen, Immer ein Ticket erstellen, Sich an Safety Standards orientieren	I1, I5

Fortgeführt auf der nächsten Seite

Lösungsansätze	**ähnliche Lösungsansätze**	**Interviewpartner**
Mit Problemen und Lücken in der Traceability ist zu rechnen, da Projekte nur von temporärer Natur sind	Links werden vernachlässigt langfristig	I3
Schritte im Tracing werden separat von spezifischen Rollen bestimmt	-	I2
Selbstbildung bei bestehenden Wissenslücken	-	I7
So tief in die Prozesse verwurzelt, dass es kein Problem darstellt	Alles wird getraced, um Sicherheitsmängel vorzubeugen, Bestehende Verfahrensmöglichkeiten	I6, I8
Tracing wird kritisch evaluiert	Tracing Prozesse und Tools werden kontinuierlich überarbeitet	I2

Kapitel 6

Diskussion der Ergebnisse

6.1 Diskussion der Forschungsfragen

In diesem Abschnitt werden auf Basis der Ergebnisse in Kapitel 5 die beiden Forschungsfragen diskutiert. Da die Forschungsfrage 2 in 3 Unteraspekte unterteilt ist, werden diese jeweils separat behandelt.

6.1.1 Beantwortung der Forschungsfrage „Was sind sicherheitsrelevante Artefakte in der Praxis?“

Wie in Kapitel 5.2.1 gezeigt, wurden im ersten Fragenblock verschiedene Artefakte von den Interviewten genannt. Die Heterogenität der Antworten verdeutlicht das unterschiedliche Verständnis von Artefakten, wie in Kapitel 5.2.1. Interviewte 3, 5, 6 und 8 verstanden Artefakte nicht nur als Softwareentwicklungsprodukte, sondern auch als physische Erzeugnisse des Hardwareentwurfs, da sie in Bereichen arbeiten, wo Hardware das Softwaredesign beeinflusst, wie z.B. Interviewter 5 in der Haushaltselektronik, wo die Software die Hardware steuert. Beispielsweise wurden Notschalter als sicherheitsrelevantes Artefakt genannt. Auch Designphase-Artefakte wie Schaltpläne wurden erwähnt. Mindestens ein Artefakt aus jedem Interview ist einer Phase des sicheren Softwareentwicklungsprozesses (zu sehen in Kapitel 2.2.3) zuzuordnen, was ein Bewusstsein für die Phasen-übergreifende Sicherheitsrelevanz zeigt.Die Verteilung der Artefakte auf die Entwicklungsphasen ist in Verteilung 2.2.3 zu sehen. Von den 44 genannten Artefakten entfallen 18 auf die Implementierungsphase, was deren Bedeutung in allen Tätigkeitsrollen der Interviewten unterstreicht. 17 Artefakte wurden den frühen Phasen (Anforderungs- und Designphase) zugeordnet, was auf die Anwendung des „Shift-Left“-Ansatzes [89], das Betrachten der Sicherheit als Eigenschaft in den früheren Phasen der Softwareentwicklung, in den Unternehmen hinweist. In den späteren Phasen wie Implementierung und Deployment wurden spezifische Sicherheitsartefakte genannt, z.B. WLAN-Verschlüsselung für Netzwerksicherheit oder Gasbrennersteuerungen für

Sicherheit von Leib und Leben. Dies zeigt die Abdeckung der Softwareentwicklungsphasen durch die Nennungen, trotz der geringen Anzahl an Interviewten. Interviewte 3, 5, 6 und 7 nannten auch domänenspezifische Artefakte, wie Geolokalisationsdaten, die nur in bestimmten Systemen relevant sind. Das Verständnis von sicherheitsrelevanten Artefakten ist daher nicht primär domänenübergreifend, sondern umfasst auch spezialisierte Domänen wie Militärtechnik, Haushaltselektronik oder Automatisierungstechnik. Zudem variierten die genannten Artefakte je nach Rolle des Interviewten: Functional Safety Manager nannten spezifische Markt- oder Kundenanforderungen, während Testmanager Negativtests erwähnten. Dies ist logisch, da die Interviewten Artefakte nennen konnten, mit denen sie beruflich vertraut sind. Um zu verorten, wie die Sicherheitsrelevanz unter den Interviewten verstanden wird, sind die Erkenntnisse in Tabelle 5.6 zu betrachten. Es zeigt sich, dass alle Interviewten Sicherheitsstandards als mögliche Richtlinien genannt haben, die Sicherheit als Eigenschaft festlegen. Diese Standards sind nicht nur einer Kategorie zuzuweisen. Interviewte 1, 2 und 6 haben beispielsweise MISRA genannt, einen Leitfaden für die Softwareentwicklung in der Automobilindustrie [63], da das von ihnen entwickelte Echtzeitbetriebssystem in Automobilen eingesetzt wird. Interviewte 3 und 4 erwähnten die RED-Richtlinie, da die Gatewaygeräte über Funkprotokolle kommunizieren und RED eine Sicherheitsrichtlinie für Funkverbindungen ist [26]. Die Domäne bestimmt somit in einigen Fällen, welche Sicherheitsrichtlinien als Artefakte betrachtet werden. Neben den domänenspezifischen Standards wurden auch domänenübergreifende Standards wie ISO 27001 oder die DSGVO erwähnt, was ihre Verbreitung und Bekanntheit unter den Interviewpartnern bestätigt. Zusätzlich finden interne Leitfäden wie Styleguides oder PEP (Prozessentwicklungsplan) Anwendung. Die Einhaltung dieser Sicherheitsstandards wird nicht nur intern evaluiert, sondern auch durch externe Prüfer wie den TÜV, was Interviewte 1, 5 und 8 erwähnten. In Fragen zu Methoden und Prozessen im Tracing bestätigten Interviewte 2, 4, 6 und 7, dass Audits durch externe Sicherheitsprüfer durchgeführt werden. Obwohl bei diesen Audits generell die Rückverfolgbarkeit überprüft wird, haben sich die Interviewten 1, 5 und 8 speziell auf die Sicherheit der Artefakte konzentriert. Laut ihnen wird im Auditprozess gezielt überprüft, ob sicherheitsrelevante Artefakte bestimmte Sicherheitseigenschaften erfüllen. Neben den Sicherheitsstandards und externen Prüfern haben 5 von 8 Interviewten angegeben, dass die gesetzten Anforderungen die Sicherheitsrelevanz bestimmen. Diese Anforderungen sind nicht nur technischer Natur, sondern umfassen auch Kundenanforderungen und marktorientierte Anforderungen. Dies zeigt, dass Sicherheit nicht nur durch technische Richtlinien bestimmt wird, sondern auch durch die Belange aller Stakeholder in der Softwareentwicklung. Interviewte 2, 3, 5 und 8, sowie implizit der Interviewte 1 durch den Verweis aus Gründen der Legacy (im Bereich der Betriebssystementwicklung), betonten, dass das

Unternehmen und insbesondere das Management die Sicherheitsrelevanz von Artefakten festlegen, vermutlich basierend auf persönlichen Erfahrungen und bekannten Standards. Interne Prozesse spielen ebenfalls eine Rolle bei der Festlegung der Sicherheitsrelevanz von Artefakten, wie von Interviewten 1, 2 und 5 erwähnt. Sicherheitsrelevante Artefakte beeinflussen die Sicherheitsrelevanz anderer Artefakte, da diese auf den Eigenschaften vorheriger Artefakte aufbauen. So können aber solche Prozesse bestimmen, wie Artefakte abhängig von den Sicherheitsstufen zu bearbeiten sind. Ein explizit genanntes Beispiel ist ASIL (Automotive Safety Integrity Level), das das Sicherheitsrisiko einer Komponente anhand der Schwere der Konsequenz und der Wahrscheinlichkeit eines sicherheitsrelevanten Vorfalls bestimmt [31]. Abhängig von den Sicherheitsstufen, wobei ASIL D die strengste und ASIL A die flexibelste ist, ändern sich die Design- und Validationsphasen der zu entwickelnden Komponenten, wodurch sich auch die Art der Artefakte ändert. Beispielsweise erfordert ASIL A keine formale Sprache für die Systembeschreibung, während ASIL D eine semiformale Sprache wie SysML benötigt [64]. ASIL wird speziell in der Automobildomäne gemäß ISO 26262 eingesetzt, doch es existieren auch andere Metriken zur Sicherheitsbewertung. Dies zeigt beispielhaft, dass Sicherheitsaspekte nicht isoliert betrachtet werden können. Die Bedürfnisse der Stakeholder beeinflussen das Management und die Teams, die die organisatorischen Sicherheitsbestimmungen festlegen, basierend auf Sicherheitsstandards und überprüft von externen Gutachtern. Dieses Muster zeigt sich bei allen Interviewten, daher gilt dies vermutlich auch bei nicht erwähnten Aspekten, aufgrund der impliziten Natur mancher Sicherheitsaspekte. Zuletzt ist auch die Diskrepanz des Sicherheitsbegriffs im Deutschen und Englischen zu beachten, wie auch schon in dem vorherigen Kapitel 2.2.1 beschrieben. In den Interviews gab es hier einen Unterschied, ob die Teilnehmer Safety oder Security dem Begriff sicherheitsrelevant zugeordnet haben. Die Interviewpartner 1, 2 und 6 haben in ihren Antworten explizit den Begriff Safety verwendet oder gemeint, dass sie sicherheitsrelevant meistens als Safety verstehen. Interviewpartner 5 hat den Begriff Safety nie erwähnt, aber die Erwähnungen von z.B. Abschaltautomatiken oder Bremssteuerungen deuten darauf hin, dass er zumindest an einigen Stellen Safety-Aspekte beschrieben haben könnte. Betrachtet man die Antworten dieser Interviewpartner grob, so sind die sicherheitsrelevanten Aspekte eher in Richtung Safety zu verstehen. Dies korreliert auch entsprechend mit den Arbeitsbereichen der jeweiligen Unternehmen, in denen sich die Interviewpartner befinden. Echtzeitbetriebssysteme, Militärtechnik oder auch in geringerem Maße Haushaltselektronik werden von dynamischen Faktoren beeinflusst, sodass auch der Safety-Aspekt eine deutlich größere Rolle spielt als bei „statischen“ Systemen, sodass es auch nicht abwegig ist, dass diese Interviewpartner mehrheitlich den Safety-Aspekt in den Vordergrund stellen. So nannte der Interviewte, wie bereits erwähnt, viele Safety relevante Artefakte, aber auch Security

relevante Artefakte wie Verschlüsselungskomponenten. Abschließend kann festgestellt werden, dass für genauere Antworten in diesem Bereich explizit die englischen Begriffe verwendet werden müssen.

Zusammengefasste Antwort der Forschungsfrage „Was sind sicherheitsrelevante Artefakte in der Softwareentwicklung?“

Sicherheitsrelevante Artefakte in der Praxis umfassen ein breites Spektrum von Hard- und Softwarekomponenten, die in verschiedenen Phasen des Software- und Hardwareentwicklungsprozesses entstehen. Aufgrund des Shift-Left-Ansatzes werden diese Artefakte mehrheitlich in der Anforderungs- und Entwurfsphase verortet, obwohl Artefakte in allen Phasen vorkommen. Die Sicherheitsrelevanz basiert auf Faktoren, die z.B. auf organisatorischen Entscheidungen, Richtlinien oder Marktanforderungen beruhen. Verschiedene Sicherheitsaspekte sind miteinander verknüpft, wobei ein Aspekt häufig von einem anderen beeinflusst wird. Darüber hinaus ist festzustellen, dass der Begriff Sicherheitsrelevanz von den meisten Teilnehmern entweder als Safety oder als Security verstanden wurde, je nach Tätigkeitsfeld des Befragten. Generell wird der Begriff Safety häufig in Bereichen wie Industrie- und Produktsicherheit verwendet, während Security eher auf den Schutz vor externen Bedrohungen in Bereichen wie Informationstechnologie und öffentliche Sicherheit bezogen wird.

6.1.2 Beantwortung der Forschungsfrage „Wird das Tracing von sicherheitsrelevanten Artefakten betrieben?“

Die erste Unterfrage untersucht, ob ein spezifisches Tracing für sicherheitsrelevante Artefakte besteht. Da alle Teilnehmer in ihren Unternehmen Tracing anwenden, ist generisches Tracing in unterschiedlichem Ausmaß vorhanden. Es stellte sich heraus, dass 4 von 8 Teilnehmern keinen Unterschied beim Tracing von sicherheitsrelevanten Artefakten machen. Drei dieser vier Teilnehmer, die Interviewten 3, 4 und 7, arbeiten in Unternehmen B. Trotz derselben Unternehmenszugehörigkeit unterscheiden sich ihre Anwendung und Einstellung zum Tracing. Interviewter 3 gab an, dass Tracelinks hauptsächlich zwischen Anforderung und Design sowie Anforderung und Tests bestehen. Als Teamleiter für die Entwicklung ist zu vermuten, dass er weniger Kontakt zum Tracing hat, da der Fokus auf Implementierungsarbeit liegt. Dies bestätigte sich, da er hauptsächlich Tracelinks in den frühen Entwicklungsphasen erwähnte. Zudem waren seine Antworten weniger umfassend im Vergleich zu den anderen Interviewten. Interviewter 4, ein Systemarchitekt (somit mehr Gestaltungsspielraum im Tracing), hatte tiefere Einblicke in den Tracingprozess. Er erklärte, dass das Tracing von sicherheitsrelevanten Artefakten in Planung sei, da es künftig

gesetzlich gefordert werde. Er sah die Traceability als Vorteil, um diese Anforderungen zu erfüllen, und identifizierte mehr mögliche Tracelinks, was zeigt, dass das Wissen zur Erweiterung vorhanden ist, aber Zeit erfordert. Er äußerte auch, dass die Entwicklungsprozesse noch nicht optimal laufen und daher Traceability derzeit nicht genügend Aufmerksamkeit bekommt. Der Interviewte 7, der dritte Mitarbeiter im Unternehmen B, hatte im Gegensatz zu den anderen beiden Interviewten unterschiedliche Ansichten zur Traceability. Er meinte, dass die Sicherheitsrelevanz innerhalb des Teams mit der Funktionalität gleichgesetzt wird. Ein wirklicher Unterschied bei den sicherheitsrelevanten Artefakten konnte somit nicht gezogen werden, sondern es wurde allgemein Stellung zur Traceability genommen, insbesondere durch Details zur Testautomatisierung, die ein direkter Teil der Tracingkette ist. Interviewter 7 präsentierte die Traceability als fortgeschrittener konzipiert, was darauf hindeutet, dass die Tracingmethoden die Belange des Testing-Teams besser erfüllen als die anderer Teams. Dies zeigt eine merkbare Diskrepanz zwischen den Abteilungen in Unternehmen B. Interviewter 8 erklärte ebenfalls, dass es keinen Unterschied im Tracing gibt. Dies kommt dadurch, da Tracing ein grundlegender Teil des Prozesses sei, hauptsächlich zur Begründung des Entwicklungsstands bei der Zertifizierung. Da die Mehrheit der Artefakte sicherheitskritisch sei, ist keine Differenzierung notwendig. Es ist zu vermuten, dass in sicherheitskritischen Domänen wie der Automatisierungstechnik Tracing ein notwendiges Kriterium ist, das sich dynamisch an den Prozess anpasst. Die anderen Interviewten, 1, 2, 5 und 6, erklärten explizit, dass Tracing dediziert für sicherheitsrelevante Artefakte durchgeführt wird, teilweise mit unterschiedlichen Methoden. Der Interviewte 1, der zusammen mit dem Interviewtem 2 im Unternehmen A arbeitet, nahm an, dass das Unternehmen A nur sicherheitsrelevante Artefakte traced. Dies liegt daran, dass die Traceability vom Safety Standard gefordert wird, um die Anforderungen des Safety Assessors zu erfüllen, der die Einhaltung des Standards überprüft. Diese Beweggründe ähneln denen von dem Interviewtem 8. Es wird explizit auf das Tracing nicht-sicherheitsrelevanter Artefakte verzichtet, da hierfür keine Notwendigkeit besteht. Interviewter 2, obwohl im selben Unternehmen tätig, behauptete etwas anderes. Er sagte, dass auch nicht-sicherheitsrelevante Artefakte getraced werden, jedoch werden sicherheitsrelevante Artefakte in einem spezialisierten Tool, JAMA, verwaltet. Wie in Unternehmen B bestehen auch hier Unterschiede zwischen den Teams. Diese Unterschiede könnten auf die verschiedenen Rollen der Teilnehmer zurückzuführen sein. Interviewter 1 nannte einen kurzen Trace Link von Hazard Risk Mitigation zu Testfall, während der Interviewte 2 den gesamten Entwicklungsprozess von der Anforderung bis zum fertigen Software-Build mit Trace Links beschrieb. Dies deutet darauf hin, dass Tracing für Interviewten 2 begleitend zur Entwicklungsarbeit erfolgt, während es für Interviewten 1 auf bestimmte Artefakte beschränkt ist, die primär zur Sicherheitszertifizierung dienen. Interviewter 5 erklärte, dass beim Tracing

von sicherheitsrelevanten Artefakten gesetzliche Bestimmungen eine Rolle spielen, die das Erstellen bestimmter Dokumente erforderlich machen. Es ist anzunehmen, dass diese wahrscheinlich der Produktzertifizierung dienen, ähnlich wie bei dem Interviewtem 8, jedoch gibt es in Unternehmen C eine explizite Unterscheidung. Die Tracingkette bei dem Interviewtem 5, ähnlich zu den anderen Teilnehmern, beginnt bei den Anforderungen und endet beim Test, wobei die Anforderungen in Haupt- und Nebenanforderungen unterteilt sind und das Produkt eine Hardwaresteuerung ist, was auf die Domäne von Unternehmen C zurückzuführen ist. Der Interviewte 6 unterschied sich in seinen Antworten deutlich von den anderen sieben Teilnehmern. Er sagte, dass die verwendeten Traceability-Werkzeuge von externen Unternehmen stammen. Dies ist ein Unterschied zu den anderen Interviewten, die ihre eigenen Lösungen verwenden. Aufgrund der sicherheitskritischen Natur der Militärindustrie, in der sich Unternehmen D befindet, sind die Entwickler wahrscheinlich verpflichtet, die Werkzeuge der Auftraggeber zu verwenden. Auf die Frage nach Unterschieden im Tracing zwischen sicherheitsrelevanten und nicht-sicherheitsrelevanten Artefakten nannte der Interviewte 6 mehrere Punkte, die auf der Echtzeitanwendungsebene eine Rolle spielen, wie die Persistenz der Fehlermeldungen, Synchronisierung der Tracingzeiten und Speicherung der Traces in einer externen Datenbank. Diese Aspekte sind im Tracing der Softwareentwicklung nicht relevant. Es ist davon auszugehen, dass der Interviewte 6 Tracing, zumindest in diesem Fragenblock, als Ablaufverfolgung der Softwareausführung verstanden hat, da seine Antworten in diesem Kontext Bedeutung haben. Das Beispiel einer Tracingkette zur Fensterüberwachung einer Raketensteuerung (beginnend von Ist-Position bis Wartung) bekräftigt diese Annahme. Somit hat der Interviewte die Differenz zwischen Echtzeit- und asynchronem Tracing gezogen.´ Zusammengefasst zeigten die vier Interviewten unterschiedliche Aussagen darüber, wie die Unterscheidung zwischen den Artefakten gemacht wird. Alle Teilnehmer nannten gewisse Vorteile des Tracings, was zeigt, dass sie in irgendeiner Weise davon profitiert haben. Sechs von acht Teilnehmern betonten die Nachvollziehbarkeit der Artefakte in verschiedenen Formen. Dies entspricht der primären Intention von Tracing und ist auch eines der in der Literatur genannten Vorteile der Traceability in der Softwareentwicklung [86]. Aus der Nachvollziehbarkeit der Artefaktbeziehungen ergeben sich weitere Vorteile wie die Abdeckung der Tests in Bezug auf Anforderungen, die Sicherheit, dass das Gesamtsystem vollständig ist, die Kondensierung der Informationen, der Schutz der eigenen Daten durch besseres Systemverständnis und Fehlerbehebung, die Prozessverbesserung durch Aufdeckung von Fehlern in der Tracingkette und die Dokumentation der Entwicklungsinformationen. Die essenzielle Rolle der Traceability in sicherheitsrelevanten Domänen zeigte sich bei den von dem Interviewtem 6 genannten Vorteilen. Tracing wird genutzt, um bei Fehlern auf die jeweiligen Traceketten verweisen zu können, um mögliche Systemfehler zu begründen oder auszuschließen. Non-

Regression-Tests, wie von dem Interviewtem 6 beschrieben, bestätigen, dass Änderungen keine neuen Fehler verursachen. Es ist anzunehmen, dass die Entwicklung sicherheitskritischer Systeme ohne Traceability nicht möglich wäre, da rechtliche und finanzielle Konsequenzen unvermeidlich wären.

Zusammengefasste Antwort der Forschungsfrage „Wird das Tracing von sicherheitsrelevanten Artefakten betrieben?“

Alle Interviewten gaben an, Tracing in unterschiedlichem Umfang zu betreiben. Die Hälfte der Befragten unterscheidet dabei zwischen dem Tracing von sicherheitsrelevanten und nicht sicherheitsrelevanten Artefakten, teilweise unter Ausschluss von generischen Artefakten. Die andere Hälfte der Befragten macht keine Unterscheidung zwischen den Artefakten und es wird vollständig getraced. Gründe dafür sind die Trennung von Funktionalität und Sicherheit, Zeitmangel, Vernachlässigung der Sicherheit oder dass die Trennung einfach nicht notwendig ist. Je nach Rolle im Unternehmen gibt es auch unterschiedliche Arten, wie das Tracing gehandhabt wird. So konzentrieren sich Entwickler auf das Tracing von Code-Artefakten, während Tester sich auf das Tracing von Tests konzentrieren.

6.1.3 Beantwortung der Forschungsfrage „Mit welchen Methoden und Prozessen wird dieses Tracing durchgeführt?“

Der zweite Unterfragenblock behandelte die Aspekte des Toolings, der Compliance und Audits, der Prozesse, der Automatisierung und der sozialen Aspekte im Tracing von sicherheitsrelevanten Artefakten. Im Tooling-Abschnitt nannten die Interviewten verschiedene Tools, die für das Tracing verwendet werden, zusammen mit dem jeweiligen Kontext des Tools. Die Tools werden grob zusammengefasst für das Anforderungsmanagement inklusive des restlichen Entwicklungszyklus, das Verwalten von Trace Links und Tracing-Metadaten, die Versionsverwaltung, Dokumentation, sicherheitsbezogene Analysen und Skripte für verschiedene Zwecke wie Tests. Das am häufigsten genannte Tool, von allen Interviewten außer dem Interviewtem 6 erwähnt, ist Jira [30]. Dies liegt daran, dass Jira in der Regel für mehrere Zwecke wie agiles Projektmanagement, Reporting und Bugtracking eingesetzt wird und flexibel an die Bedürfnisse des Unternehmens angepasst werden kann. Die Tracingfunktionalität von Jira beruht auf der Vielseitigkeit des Tools, sodass angenommen werden kann, dass Jira die Anforderungen von vier der fünf befragten Unternehmen erfüllt. Ein weiteres häufig genanntes Tool ist Polarion [81]. Dieses wurde von allen drei Interviewten (3, 4 und 7) im Unternehmen B genannt, was die Häufigkeit erhöht. Polarion ist ein proprietäres Tool, das den

gesamten Entwicklungsprozess einschließlich Tracing gezielt verwalten kann. Andere erwähnte Tools wie Enterprise Architect oder IBM Doors heben sich durch zusätzliche Funktionalitäten von anderen Tools ab, verfolgen jedoch im Wesentlichen das gleiche Ziel, eine gezielte Verbindung zwischen allen Entwicklungsphasen zu ziehen. Auffällig ist das Tool JAMA, das von den Interviewten im Unternehmen A erwähnt wurde und dediziert für sicherheitsrelevante Artefakte genutzt wird. Aus den Erklärungen der Interviewten, einschließlich dem Interviewtem 4, geht nicht genau hervor, warum dies zutrifft, da die genannten Funktionalitäten sich nicht wesentlich von den anderen Tools unterscheiden. Nach genauerer Recherche stellt sich heraus, dass JAMA ISO 26262 zertifiziert ist [45], welches in Unternehmen A Anwendung findet. Somit zeigt sich, dass bestimmte Standards die Nutzung von Tools einschränken und einige Artefakte in solchen Tools verwaltet und bearbeitet werden müssen. Weitere genannte Tools verfolgen individuelle, eingeschränkte Zwecke (z.B. Skripte für die Artefaktgenerierung) oder werden nur durch die Tracingkette in den Tracingprozess einbezogen (z.B. Wireshark für die Überwachung des Netzwerkverkehrs). Der nächste Fragenblock thematisierte die Compliance- und Audit-Aspekte des Tracings. Die Erkenntnisse hierzu ergänzen die Ergebnisse der Forschungsfrage 1, daher werden in diesem Abschnitt nur die neuen Erkenntnisse genannt. Im Allgemeinen gingen die Interviewten nicht sehr ins Detail und konnten auch nicht viele neue Erkenntnisse liefern. Der Interviewte 3 konnte beispielsweise keine Aussagen zu Compliance machen, da diese keine Rolle spielen. Vermutlich hat das jeweilige Team keinen Bezug zu den Standards und das Tracing wird nur rein methodisch angewendet. Diese Annahme wird auch durch den Interviewten 7 bestätigt, der erklärte, dass Systemarchitekten und Requirements-Ingenieure die Standards auflösen und in Anforderungen an die jeweiligen Tester und Entwickler übergeben. Der Interviewte 4, ein Systemarchitekt, konnte ebenfalls keine Details nennen, da Nachweisbarkeit und Dokumentation erst in der Zukunft gefordert werden. Daher ist zu vermuten, dass in Unternehmen B bezüglich der Compliance-Richtlinien im Kontext des Tracings keine unternehmensweite Transparenz besteht und zum Zeitpunkt des Interviews auch keine Notwendigkeit bestand. Der Interviewte 6 betrachtete Compliance zusätzlich im Kontext des Anwendungstracings. Neben der Nachvollziehbarkeit des gesamten Entwicklungsprozesses werden sicherheitsrelevante „Artefakte“ im Applikationsverlauf in Echtzeit protokolliert. Diese Applikationstraces werden bis in die Anforderungsebene zurückverfolgt, um zu überprüfen, ob mögliche Ereignisse im Systemverlauf ein berechtigtes Auftreten haben. Damit sind im Punkt der Compliance die beiden Tracingebenen verbunden und werden gleichwertig betrachtet. Neben den Applikationstraces gibt es zudem die FMEA-Untersuchungen, die wie auch schon andere Audits von einer dritten unabhängigen Instanz überprüft werden [32]. Diese FMEA (Failure Mode and Effects Analysis) Untersuchungen dienen dazu, Safety-Fehler in einem System qualitativ

aufzudecken und gegebenenfalls frühzeitig zu beheben. Gleichzeitig können sie dazu dienen, spätere Fehler durch Kategorisierung und Einschätzung auf Basis vergangener Fehler besser zu beurteilen. Eine ähnliche Rolle spielen die HARA-Analysen, wie die Interviewten 2 und 1 in früheren Fragenblöcken aussagten [22]. Im Gegensatz zur FMEA-Analyse geht es bei der HARA um die Analyse bestehender Risiken. Diese und viele weitere nicht genannte Analysemethoden zur Fehleranalyse, Einordnung und Behebung bieten eine Möglichkeit, Compliance-Richtlinien hinsichtlich Safety-Aspekten (im Gegensatz zu Security) zu erfüllen. Dies gilt insbesondere, da diese Analysemethoden nur von Interviewten genannt wurden, die in Safety-Domänen arbeiten. Zu den Audits wurden wenig zusätzliche neuartige Aussagen gemacht, außer dass sie den Zweck haben, die Einhaltung von Safety- und Security-Standards zu überprüfen. Dies liegt vermutlich daran, dass Audits, wie indirekt von allen gesagt, mehrheitlich (außer dem internen Auditing, das von Interviewtem 1 erwähnt wurde) von externen Stellen durchgeführt werden und daher nicht von den Entwicklern und Testern ausgeführt werden. Nach den Aussagen von den Interviewten 2 und 8 haben Audits in Bezug auf Traceability den positiven Nebeneffekt, dass Entwickler, die eine gewisse Distanz zu den Sicherheitsstandards haben, durch den Auditprozess mehr in Kontakt mit den Richtlinien kommen, da Fehler und Verstöße gegen die Standards transparent aufgedeckt werden. Der Interviewte 5 äußerte zudem, dass in der Haushaltselektronik-Domäne die Audits dazu führen können, dass bei Fehlern die gesamte Hardware neu konstruiert wird. Auch wenn dies von den Interviewten aus den Safety-Domänen nicht geäußert wurde, gilt dies vermutlich generell in der Safety-Domäne, da Fehler in der Praxis mit Personen- und Sachschäden verbunden sind und der Auditprozess der letzte Schritt vor der Markteinführung des Endprodukts ist. Somit ist anzunehmen, dass der Auditprozess in sicherheitskritischen Domänen auch eine Überprüfung auf Markttauglichkeit des Endprodukts beinhaltet. Der nächste Aspekt behandelt die Prozessintegration des Tracings sicherheitsrelevanter Artefakte. Hierbei ging es um die Integration in den allgemeinen Entwicklungsprozess und den Umgang mit Sicherheitsaspekten, da diese in dem Kontext eine wichtige Rolle spielen. Die Ergebnisse variieren stark zwischen den Unternehmen, anders als andere Aspekte des Tracing, was vermutlich auf die unterschiedlichen Entwicklungsprozesse zurückzuführen ist. Dennoch gibt es Gemeinsamkeiten: Alle Interviewten außer Interviewtem 3 bestätigten die Existenz von Werkzeugen, Methodiken oder Schritten zur Integration des Tracings.Dies hat sich in der Tiefe der Details unterschieden. Interviewter 4 beschrieb allgemein, dass es Methodiken und Schritte gibt, ging jedoch nicht ins Detail, da dies von Unternehmen zu Unternehmen unterschiedlich ist.Der Interviewte 2 ging hingegen mehr ins Detail. Er erklärte, dass der Commit von Entwicklungsmaterial generell an den jeweiligen Jira Issue angebunden ist, wodurch ein Trace Link von Anfang an besteht. Sicherheitsrelevante Artefakte werden zusätzlich an das JAMA-

Projekt angebunden, das Verweise auf sicherheitsrelevante Anforderungen erstellt. Auch in der Qualitätssicherung wird Tracing integriert. Interviewter 6 beschrieb, dass Traceability genutzt wird, um Änderungen anhand der Tracingkette nachzuverfolgen und Prüfabläufe durch Non-Regression-Tests anzupassen. Nur der Interviewte 1 und 2 konnten spezifische Aussagen zum sicheren Softwareentwicklungszyklus machen, da bei anderen Interviewten keine Trennung im Zyklus existierte oder das Konzept unbekannt war. Der Interviewte 1 nannte konkret, dass es ein Qualitätsmanagementsystem gibt, das zahlreiche Anweisungen für sichere Softwareentwicklungszyklen enthält. Es definiert auch, wie das Tracing aussieht und welche Tools eingesetzt werden. Die praktische Umsetzung dieses Zyklus zeigt sich in den Aussagen von Interviewtem 2. Der Interviewte 2 ergänzte, dass statische Codeüberprüfungen mit festen Regelsätzen durchgeführt werden müssen, was in den Tracingprozess eingebunden ist. Tracing für sicherheitsrelevante Artefakte unterliegt somit strikteren Regeln, die manuelle Schritte oder zusätzliche Sicherheitsüberprüfungen beinhalten können. Insgesamt fiel auf, dass außer den Interviewten 1, 2 und 6 keine praktischen Beispiele genannt wurden. Dies liegt vermutlich daran, dass genauere Antworten ein umfassendes Wissen der Prozessabläufe erfordern. Solches Wissen ist förderlich, wenn die entsprechende Tätigkeit passend ist oder langjährige Erfahrung im Prozess besteht, was bei den Interviewten 1, 2 und 6 der Fall ist. Im vierten Fragenblock wurden Fragen zum Automatisierungsstand im Tracing in den Firmen gestellt. Die Interviewten 3 und 4 gaben an, dass in ihren Unternehmen keine Automatisierung für das Tracing existiert. Gründe sind, dass bestehende Automatisierungsmöglichkeiten die Bedürfnisse und Aufgaben der Teams nicht angemessen erfüllen, insbesondere den interpretativen Teil der Entwicklung. Vermutlich ist die Anzahl der zu tracenden Artefakte gering (ausgehend von den vorherigen Ergebnissen), sodass die manuelle Arbeit toleriert wird. Trotzdem führt dies laut dem Interviewtem 3 dazu, dass das Tracing fehlerhaft oder gar nicht durchgeführt wird, was Lücken in den Tracingketten verursacht und die fehlende Automatisierung die Arbeit erschwert. Skepsis oder Abneigung gegenüber Automatisierung besteht jedoch nicht, da an den Entwicklungsprozessen, einschließlich des Tracings, gearbeitet wird. Die übrigen Interviewten 1, 2, 5, 6, 7 und 8 berichteten, dass das Tracing teilweise automatisiert ist. Die Automatisierung findet jedoch nur eingeschränkt in ein oder zwei Entwicklungsphasen statt, etwa bei der Überprüfung der Testabdeckung oder der automatischen Integration von Codekomponenten. Zudem beschränken sich die genannten Automatisierungen auf die jeweiligen Tätigkeitsfelder der Interviewten. Der Testmanager konnte nur Automatisierungen für Tests angeben, und der Entwickler nur für die Integration. Ob es automatisierte Schritte in anderen Entwicklungsabschnitten innerhalb der Unternehmen gibt, konnte daher nicht beantwortet werden. Automatisierung basierend auf Künstlicher Intelligenz fehlt völlig, was auf die Komplexität des interpretativen Teils der Softwareentwicklung zurückzuführen ist, wie

der Interviewte 4 anmerkte. Besonders in der Safety-Entwicklung müssen laut dem Interviewtem 1 False Positives vermieden werden, da Korrektheit entscheidend ist. Weitere Automatisierungsgründe stimmen mit den Aussagen von Interviewten 3 und 4 überein. Ergänzend wurde geäußert, dass bestehende Automatisierungstools und -methodiken hinderlich sind, weshalb bei Einführung neuer Automatisierungen andere bestehende Komponenten verändert werden müssten und erhöhter Mehraufwand entsteht. Abgesehen von Mehraufwand, Abhängigkeit von externen Tools und Fehleranfälligkeit wurden keine weiteren Nachteile genannt. Mit der Weiterentwicklung der Künstlichen Intelligenz könnten diese Nachteile tolerierbar sein, ohne die Funktionalität des Tracings zu beeinträchtigen. Dies beweisen auch die genannten Vorteile der Automatisierung, da die Interviewten aussagen, dass die manuelle Arbeit reduziert wird und auch die manuell erzeugten Fehler wegfallen. Der letzte Teil des Fragenblocks befasste sich mit der sozialen Teilhabe im Tracingprozess und den Rollenunterschieden. Es zeigte sich, dass direkte und kurze Kommunikation im Tracing notwendig ist, da fehlende Kommunikation zu Fehlern oder Lücken führen kann. Da Tracing verschiedene Entwicklungsphasen und -abteilungen verbindet, fördert es die Übermittlung von Domänenwissen. In den Teams der Interviewten gibt es verschiedene Methoden und Prozesse, um die Kommunikation zu erleichtern, wie Meetings, Schulungen oder kollaborative Tools. Die Interviewten 2, 3 und 5 erwähnten, dass Rollen an unterschiedlichen Stellen der Entwicklung sitzen und mit verschiedenen Artefakten und Trace Links arbeiten. Für das Tracing selbst sind bestimmte Rollen entscheidend. Softwarearchitekten übernehmen Schlüsseltätigkeiten wie das Setzen und die Instandhaltung der Tracinglinks sowie das Verwalten der Anforderungen. In Safety-Domänen muss der Functional Safety Manager das Projektmanagement und die Validierung des Tracings übernehmen, wie von dem Interviewtem 1 angegeben. Da keine weiteren Antworten zu speziellen Rollen existieren, ist anzunehmen, dass andere Rollen ohne Verwaltungsbefugnisse im Tracing keine besonderen Funktionen erfüllen. Die Verteilung der Befugnisse unterscheidet sich daher im Wesentlichen nicht von anderen Tätigkeiten in der Softwareentwicklung, basierend auf den Interviewergebnissen.

Zusammengefasste Antwort der Forschungsfrage „Mit welchen Methoden und Prozessen wird dieses Tracing durchgeführt?“

In der Untersuchung zum Tracing von sicherheitsrelevanten Artefakten wurden verschiedene Methoden und Prozesse identifiziert, die in Unternehmen unterschiedlich implementiert werden. Zu den hervorgehobenen Tools gehören Jira, Polarion, Enterprise Architect, IBM Doors und JAMA. Jira wird besonders häufig für das Projektmanagement genutzt, weswegen es von 7 von 8 Interviewten genannt wurde und im Tracing im Vordergrund steht. Speziell für sicherheitsrelevante Artefakte werden Tools wie JAMA genutzt, da diese zertifiziert sind für bestimmte Sicherheitsstandards wie der ISO 26262. In Bezug auf die Prozessintegration des Tracings in den Softwareentwicklungszyklus bestätigten die meisten Interviewten, dass spezifische Werkzeuge und Methodiken verwendet werden. Dies umfasst das Anbinden von Entwicklungsaktivitäten an Jira Issues, die Integration von sicherheitsrelevanten Artefakten über JAMA und die Einbindung von Tracing in Qualitätssicherungsmaßnahmen, wie Non-Regression Testing. Automatisierte Tracing-Schritte sind jedoch begrenzt und häufig nur innerhalb einzelner Entwicklungsphasen möglich. Compliance und Audits wurden ebenfalls thematisiert, wobei nur wenige neue Erkenntnisse genannt wurden. Einige Interviewte berichteten, dass Compliance-Richtlinien und Audits helfen, Sicherheitsstandards durch Transparenz und Fehleraufdeckung stärker zu integrieren. Der Automatisierungsgrad im Tracing ist generell gering, wobei einige Unternehmen Teilaspekte automatisiert haben. Die Herausforderungen liegen in der Komplexität und Fehleranfälligkeit solcher Systeme, was durch zukünftige Fortschritte in der künstlichen Intelligenz möglicherweise verbessert werden kann. Sozial gesehen ist die direkte Kommunikation im Tracingprozess von hoher Bedeutung, um Fehlinterpretationen und Lücken zu vermeiden. Rollenspezifisch sind insbesondere Softwarearchitekten und Functional Safety Manager entscheidend für die Durchführung und Überwachung des Tracings.

6.1.4 Beantwortung der Forschungsfrage „Welche Probleme bestehen bei dem Tracing von sicherheitsrelevanten Artefakten?“

Zuletzt wurden den Interviewten Fragen zu den Nachteilen und Problemen des Tracings gestellt und wie die Unternehmen mit diesen Problemen umgehen. Auf die Frage nach den Nachteilen durch Tracing von sicherheitsrelevanten Artefakten konnten fast alle Interviewten entweder keine nennen oder erwähnten lediglich den erhöhten Aufwand. Der Interviewte 7 konkre-

tisierte diesen Aufwand und erklärte, dass Schulungen externer Firmen, mit denen er zusammenarbeitet oder gearbeitet hat, notwendig sind, um deren Bewusstsein für Sicherheit und Traceability zu schärfen. Dies deutet darauf hin, dass die Durchführung des Tracings von Externen aufgrund des erhöhten Aufwands oder mangelnden Bewusstseins für Tracing vernachlässigt wird. Interviewter 3 nannte als Nachteil, dass bei manuellem Tracing die Aktualität der Tracinginformationen nicht sichergestellt ist. Dieser Nachteil entsteht durch die Vernachlässigung des Tracings aufgrund des erhöhten Aufwands und ist daher kein inhärenter Nachteil des Tracings selbst. Auch der Mehraufwand als Nachteil ist kein eigentlicher Nachteil der Traceability, sondern etwas, das zusätzlich zur Entwicklungsarbeit getan werden muss und diese behindert. Dies ist jedoch auch bei anderen „zusätzlichen“ Entwicklungsarbeiten der Fall, wie dem Kommentieren oder Testen von Code, ohne den Sinn und Zweck der Tätigkeit abzuwerten. Dementsprechend haben die Interviewten keine echten Nachteile nennen können, was zeigt, dass Tracing von ihnen grundsätzlich als positiv aufgenommen wird. Im Gegensatz zu den Nachteilen wurden jedoch diverse Probleme und Herausforderungen bei der Durchführung des Tracings genannt. Ein Teil dieser Probleme ergibt sich aus dem manuellen Tracing, bedingt durch die Fehleranfälligkeit, fehlende Aktualität der Links, benötigte Kenntnisse und die notwendige Kooperation aller Akteure in der Softwareentwicklung. Diese Punkte wurden bereits von dem Interviewtem 3 als Nachteil benannt. Zudem ist der erhöhte Aufwand, sei es zeitlich oder durch die Komplexität, ein großer Hauptfaktor für das Bestehen dieser Probleme, was auch von den Interviewten 6 und 8 bestätigt wurde. Diese beiden konnten keine wirklichen Herausforderungen identifizieren, da Tracing stark in den Prozess integriert ist und dessen fehlende Durchführung das Vorankommen der Entwicklung beeinträchtigt. Der erhöhte Zeitaufwand und die Komplexität wurden somit nicht als Problem wahrgenommen. Ein weiteres genanntes Problem ist das Tooling. Einerseits gibt es Schwierigkeiten beim Aufsetzen automatisierter Tools, was wieder auf die bereits erwähnte generelle Komplexität zurückzuführen ist. Andererseits wurde die Zusammenarbeit verschiedener Tools von dem Interviewtem 2 als Problem identifiziert, ein Aspekt, der auch in der Literatur erwähnt wird [24]. Dies ist logischerweise ein Problem in Unternehmen, die viele Tools für das Tracing einsetzen, wie es in Unternehmen A der Fall ist. Der Interviewte 7 erwähnte zudem die Verfügbarkeit als Problem, da die Testautomatisierungs-Software konsistent auf einem Rechner laufen muss. Da dies ein allgemeines Verfügbarkeitsproblem ist und die Lösung in der Bereitstellung mehrerer Rechenpunkte liegt, ist dies kein spezifisches Problem der Tracing-Tools. Die Frage zu den Herausforderungen wurde durch eine Anschlussfrage spezifiziert, welche Sicherheitsrisiken und -herausforderungen beim Tracing berücksichtigt werden müssen, da keiner der Interviewten sicherheitsrelevante Aspekte genannt hatte. Die Interviewten 2, 3, 7 und 8 konnten hierzu nichts nennen oder verwiesen auf bestehende Probleme wie

nicht aktualisierte Trace Links. Dies waren auch die Teilnehmer, die keine Differenz zwischen den Artefakten ziehen, weshalb vermutlich auch keine Risiken diesbezüglich genannt wurden. Die restlichen Interviewten haben indirekt oder direkt auf die Tracingdaten als potenziell sicherheitsrelevantes Artefakt hingewiesen. Tracingdaten können aufgrund ihrer Sensibilität oder fehlenden Aktualität von Angreifern missbraucht oder durch gesetzliche Bestimmungen wie dem Datenschutz gelöscht werden. Die Interviewten deuteten hier implizit auf die Wahrung der Vertraulichkeit, Integrität und Authentizität der Tracingdaten hin, was in Kapitel 2.2.1 als wichtige Sicherheitseigenschaften erwähnt wurden. Somit wird durch unsere Interviewten bestätigt, dass auch beim Tracing grundlegende IT-Sicherheitsaspekte eine Rolle spielen. Zuletzt wurde gefragt, wie die Interviewten in ihren Teams oder Unternehmen mit Herausforderungen umgehen, die beim Tracing von sicherheitsrelevanten Artefakten auftreten. Die Interviewten 6 und 8 konnten keine wirklichen Herausforderungen identifizieren, da Tracing stark in den Prozess integriert ist. Das Fehlen von Tracing würde das Vorankommen des subjektiv wahrgenommenen Entwicklungserfolgs beeinträchtigen. Daher wurden erhöhter Zeitaufwand, Komplexität und diverse andere Herausforderungen nicht als Problem identifiziert. Vier der sechs restlichen Interviewten sagten zusammengefasst aus, dass effektives Tracing gute Prozessplanung erfordert, um spätere Probleme optimal bewältigen zu können. Diese generelle Aussage ähnelt den Aussagen von den Interviewten 6 und 8, da auch hier Tracing als integraler Bestandteil des Prozesses verstanden wird. Der Interviewte 7 nannte Weiterbildung als mögliche Lösung und setzte die Herausforderungen im Tracing damit indirekt mit allgemeinen Herausforderungen gleich. Einzig der Interviewte 3 vertrat den Standpunkt, dass Probleme zu erwarten und anzunehmen sind. Diese Haltung beruht darauf, dass die Projekte im Team von dem Interviewtem 3 externer Natur und kurzfristig sind. Daher lohnen sich langfristige Prozessplanungen für das Tracing nicht. Auf Basis dieser Erkenntnisse kann abgeleitet werden, dass das Tracing von sicherheitsrelevanten Artefakten in die allgemeine Entwicklungsprozessplanung eingebunden ist. Folglich wird angenommen, dass bei auftretenden Herausforderungen methodisch fundiert reagiert wird.

Zusammengefasste Antwort der Forschungsfrage „Welche Probleme bestehen bei dem Tracing von sicherheitsrelevanten Artefakten?“

Die meisten Interviewten berichteten, dass sie keine spezifischen Nachteile des Tracings identifizieren konnten, außer einem erhöhten Aufwand, der hauptsächlich mit Schulungen und der Implementierung von Traceability zusammenhängt, und somit nicht dem Tracing an sich zuzuordnen sind. Dieser erhöhte Aufwand führt dazu, dass die Teilnehmer Herausforderungen bei der Verwaltung der Trace Links sehen und dem Tooling. Sicherheitsrisiken wurden bei den Tracingdaten selber verortet, um diese zu wahren, müssen die bekannten Sicherheitsaspekte gewährleistet werden. So gut wie alle Interviewten behandeln die Probleme durch die Einbindung des Tracings in die Prozessplanung und durch Erwerb von Kenntnissen.

6.2 Handlungsempfehlungen für das Tracing von sicherheitsrelevanten Artefakten

Modelle haben grundlegend 4 Kriterien, die die Charakterisierung und das Verständnis des Modells definieren [72].

Kriterium 1:
Was ist das Original (des Modells)?

Um die Ergebnisse der Arbeit zu verwerten, werden Handlungsempfehlungen geäußert. Dies passiert auf Basis der Probleme und positiv zu wertenden Aspekte des ermittelten Tracings, die von den Interviewten vermittelt wurden und die darauf aufbauende Diskussion.

Kriterium 2:
Für wen ist das Modell?

Die Zielgruppe für die Handlungsempfehlungen sind Studierende sowie Berufseinsteigerinnen und Berufseinsteiger, die für das jeweilige Themenfeld einen grundlegenden Einstieg benötigen. Dies ist die geeignetste Zielgruppe, da die Ergebnisse und Diskussionen nicht zu detailliert sind, jedoch durch die Beantwortung der Forschungsfragen einen generellen Überblick über das Themenfeld darstellen.

Kriterium 3:
Zu welchem Zweck dienst das Modell?

Die Handlungsempfehlungen sollen dazu dienen, mögliche Ideen und Anreize für die praktische Durchführung von Tracing zu motivieren, um bestehende Missstände der Praxis zu reduzieren oder zu verhindern.

Kriterium 4:
Welche Eigenschaften (des Originals) sind dafür relevant?

Um Anhaltspunkte für die Handlungsempfehlungen zu finden, wurden häufige Probleme und besonders gut funktionierende Prozesse und Methoden, die in den Ergebnissen und Diskussion thematisiert wurden, betrachtet. Hierfür wurden die sechs wichtigsten Aspekte herausgesucht. Aufgrund der thematischen Überschneidungen der behandelten Aspekte werden weitere potenzielle Handlungsempfehlungen nicht erwähnt, um eine redundante Darstellung von Informationen zu vermeiden.

6.2.1 Bewusstsein für Sicherheitsrelevanz durch Sicherheitsstandards schaffen

In den geführten Interviews wurde geäußert, dass Sicherheits- und Security-Standards in unterschiedlichem Maße bewusst wahrgenommen werden. Diese legen präzise und klar fest, wann ein Artefakt als sicherheitsrelevant einzustufen ist und wann nicht. Zudem sind sie nicht beschränkt auf individuelle Gegebenheiten und Erfahrungswerte der einzelnen Akteure in der Softwareentwicklung. Da jedoch eine detaillierte Beschreibung der Sicherheitsaspekte in den jeweiligen Standards durch die interviewten Akteure nur unzureichend erfolgte, sollte eine Sicherstellung des Bewusstseins und der einheitlichen Wahrnehmung der Standards erfolgen. Dies kann beispielsweise mittels der Projektverwaltungstools erfolgen, indem zusätzliche Informationen den Artefaktbeschreibungen beigefügt werden, die auf Teile der relevanten Standards hinweisen und den jeweiligen Projektbearbeitern bei späteren Ungewissheiten als Grundlage dienen können. Des Weiteren kann in Meetings und Präsentationen, die im Rahmen von Softwareentwicklungsprojekten stattfinden, auf relevante Standardrichtlinien hingewiesen und informiert werden, um deren Vernachlässigung zu vermeiden.

6.2.2 Externe/Dienstleister müssen dem Tracingprozess folgen

Die Interviewpartner I1, I4 und I6 berichteten von Problemen bei der Einbindung Externer in das Tracing. Die Probleme lagen hier in Zweifeln an der Sinnhaftigkeit des Tracings, in erhöhten Kosten und Aufwänden, die bei der Einbeziehung in das Tracing zu erwarten sind und auch insbesondere in Sicherheitsbedenken. Bei einer steigenden Anzahl von Projekten mit Externen muss daher auch berücksichtigt werden, dass bestehende und zukünftige externe Mitarbeiter aktiv in den Tracingprozess eingebunden werden können. Dazu wäre ein erster Schritt, Externen einen schnellen und unkomplizierten Zugang zu den Tracingtools zu ermöglichen. Dies könnte über Gastaccounts mit definierten Rollen und Zugriffsrechten auf

die jeweiligen Tracingfunktionalitäten realisiert werden. Zusätzlich könnte eine Dokumentation erstellt werden, die die jeweiligen Prozesse im Detail definiert und verständlich erklärt. Bestehende Vorbehalte gegenüber dem Tracing können dadurch abgebaut werden, dass den externen Partnern aktiv aufgezeigt wird, welche Vorteile das Tracing bietet und auch an praktischen Beispielen verdeutlicht wird, welche Konsequenzen eine Vernachlässigung hat. Im Hinblick auf sicherheitsrelevante Aspekte, insbesondere unter Einbeziehung sicherheitsrelevanter Artefakte, sollten vertragliche Vereinbarungen getroffen werden, welche Sicherheitsrichtlinien bei der Zusammenarbeit wann und in welchem Kontext gelten (unter Berücksichtigung der Belange beider Unternehmenspartner) und speziell für das Tracing ist zu prüfen, in welchem Umfang und auf welche Weise Externe Einblick in unternehmensinterne Daten erhalten.

6.2.3 Automatisierte Komponenten im Tracing implementieren

Die interviewten Personen berichteten von Fällen gänzlicher fehlender Automatisierung (wie im Falle des Unternehmens B) oder teilweise fehlender Automatisierung. Viele der von den Interviewten identifizierten Probleme sind auf das manuelle Instandhalten der Tracinglinks zurückzuführen, da dies vernachlässigt wird oder falsch ausgeführt wird. In diesem Kontext kann die Orientierung an den Prozessen des Unternehmens A als empfehlenswert erachtet werden. Diesbezüglich ist das Setzen von Tracing-Informationen bei Änderungen des Artefaktbestandes seitens der verwendeten Projektverwaltungstools zwingend zu fordern. Um die jeweiligen Tracelinks korrekt zu setzen und aktuell zu halten, können maschinelle Lernverfahren auf periodischen Zeitabständen eingesetzt werden. Diese durchlaufen den Entwicklungsstand der Teams und geben Vorschläge, welche Trace-Links gesetzt oder geändert werden müssen. Dies gewährleistet, dass die Entscheidungsgewalt, ob ein Tracing-Link gesetzt werden muss, noch beim Mitarbeiter liegt, wobei sich die manuelle Arbeit dennoch zu gewissen Teilen reduziert.

6.2.4 Traceability von sicherheitsrelevanten Artefakten verpflichtend machen

In den geführten Interviews wurde ersichtlich, dass das Tracing effektiv dazu beitragen kann, die Sicherheit zu erhöhen, dass die Tragweite einer Änderung auf das beschränkt bleibt, was mit der Änderung intendiert wurde. Dies wurde insbesondere bei den Unternehmen A und C ersichtlich, da diese in Domänen arbeiten, in denen eine falsche Änderung finanzielle Auswirkungen und Schäden am Leib und Leben haben kann. Für Unternehmen, die sich neu gründen oder ihre Tätigkeitsdomäne ausweiten und mit einer erhöhten Anzahl an sicherheitsrelevanten Artefakten arbeiten, sollte die gesetzliche

Verpflichtung bestehen, Tracing für alle Artefakte durchzuführen. Die gesetzliche Verpflichtung sollte jedoch nicht nur die Durchführung, sondern auch die geeignete Nutzung zertifizierter Tools, die Schulung von Mitarbeitern hinsichtlich der Durchführung von Tracing sowie die Durchführung von Audits in bestimmten Zeitabständen umfassen. Letztere dienen der Information über Missstände sowie der Motivation zur korrekten Einhaltung von Traceability-Richtlinien. Des Weiteren könnte die Einführung zusätzlicher Rollenerweiterungen in Erwägung gezogen werden, die eine gleichwertige Funktion wie die des Datenschutzbeauftragten beim Datenschutz erfüllen. Diese Funktion umfasst die Überprüfung der korrekten Einhaltung des Tracings sowie die Rolle als Ansprechperson für die Mitarbeiter.

6.2.5 Interdisziplinäre Arbeit und Kommunikation während des Tracings fördern

Die Auswertung der Interviews hat ergeben, dass das Tracing in verschiedenen Abteilungen der Softwareentwicklung bearbeitet wird. Dabei wurden bei Unternehmen A und Unternehmen B teilweise unterschiedliche Gegebenheiten über das Tooling, die Schritte, die zum Tracing benötigt werden, die Art und Weise, wie Tracing verstanden wird, sowie die Ausmaße des Tracings berichtet. Gleichzeitig wurde von den Teilnehmern ausgesagt, dass eine effiziente und korrekte Kommunikation zwischen den Teams von essenzieller Bedeutung ist, um die Zusammenhänge zwischen den Tracelinks nachvollziehbar zu gestalten. Es ist daher von entscheidender Wichtigkeit, dass die Teams auf allen Ebenen aufeinander abgestimmt sind. Es ist von entscheidender Bedeutung, dass allen Mitarbeitern in den verschiedenen Teams (und externen Arbeitern) bewusst gemacht wird, dass alle Artefakte ausreichend Informationen beinhalten müssen, damit Mitarbeiter mit anderen Tätigkeitsrollen keine Schwierigkeiten haben, den Hintergrund und den Einsatzkontext des fremden Artefakts zu verstehen. Es sollte ein einheitliches Format für die Beschreibung von Artefakten und für die Erstellung von Trace-Links innerhalb des Tools festgelegt werden. Eigene Lösungen innerhalb der Teams sollten dabei lediglich auf Tätigkeiten begrenzt werden, die ausschließlich innerhalb der Teams von Relevanz sind. Darüber hinaus ist es von entscheidender Bedeutung, eine offene Gesprächskultur zwischen den Teams zu etablieren. Auf diese Weise können Unstimmigkeiten zeitnah ausgeräumt werden, wodurch fehlerhafte Tracelinks aufgrund von Missverständnissen verhindert werden können.

6.2.6 Zugriff auf sicherheitsrelevante Tracingdaten vor Unbefugten schützen

Einige der Befragten äußerten, dass ein gewisses Sicherheitsrisiko darin besteht, dass Tracingdaten unbefugt eingelesen und missbraucht werden

können. Der böswillige Zugriff auf diese Daten ist von besonderer Schwere, da mittels der Tracingdaten sich ein Überblick über die Software bis ins letzte Detail gemacht werden kann. Bei einer guten Traceability werden die Interaktionen aller Komponenten zueinander klar und deutlich erfasst. Gleichzeitig besteht die Möglichkeit, aus den Tracingdaten heraus Einblick in die internen Entwicklungsprozesse zu gewinnen. Aus diesem Grund ist es unerlässlich, Sicherheitsmaßnahmen bezüglich der Tracingdaten zu implementieren. Der Zugriff auf diese Daten sollte ausschließlich befugten Personen gestattet werden. Dies kann durch die Einschränkung des Zugriffs auf die Daten innerhalb des jeweiligen Tools erreicht werden. Die Speicherung der Tracingdaten sollte zudem ausschließlich in den jeweiligen Tools erfolgen und nicht über jegliche Kommunikationsmedien an andere Mitarbeiter weitergegeben werden, da dies das Risiko birgt, dass die Daten an Unbefugte weitergegeben werden. Falls die Daten zwischen Tools übertragen werden, ist eine Verschlüsselung der Daten erforderlich. Mitarbeiter, die Zugriff auf diese Daten erhalten, sollten durch zusätzliche Schulungen oder Meetings bezüglich des Themas sensibilisiert werden, um einen verantwortungsbewussten Umgang mit den jeweiligen Informationen zu gewährleisten. Dies kann gefördert werden, indem in den jeweiligen Tools die Mitarbeiter nochmals dediziert darauf hingewiesen werden.

6.3 Validität der Ergebnisse

Um die Grenzen dieser Arbeit zu ermitteln, müssen wir die wissenschaftliche Gültigkeit dieser Arbeit evaluieren. Um hier methodisch voranzugehen und nach wissenschaftlichen Maßstäben zu beurteilen, überprüfen wir die Konstruktvalidität, Interne Validität, Externe Validität und Reliabilität [92].

6.3.1 Konstruktvalidität

Konstruktvalidität prüft, ob die Interviewfragen in unserer Studie wirklich das überprüfen, was wir untersuchen möchten. Es geht darum, sicherzustellen, dass unsere Methoden passend sind, um unsere Forschungsfragen zu beantworten [92]. Die Interviewfragen sind so ausgelegt, dass sie speziell darauf ausgerichtet sind, die Forschungsfragen zu beantworten. Hierfür wurden zu den Hauptfragen mehrere optionale Fragen gestellt, die ein Beispiel vom Interviewten gefordert haben. Es wurde angenommen, dass dadurch, dass die Interviewten alle das Tracing aus der praktischen Sicht wahrnehmen, mittels Beispielen mehr Aussagen treffen können. Vor der eigentlichen Durchführung des Interviews wurde wie schon vorher beschrieben ein Testdurchlauf mit einer Person durchgeführt, die ein theoretisches Wissensfundament in dem Thema besitzt und somit einen hypothetischen Interviewverlauf simulieren konnte. Die inhaltlichen Schwächen dieses Testlaufs wurden optimiert und die Fragen angepasst. Zudem wurde versucht, durch die zwei

Einstiegsfragen und eigener Definition zum Verständnis von Traceability und Sicherheitsrelevanz von Artefakten Unverständlichkeiten aufzulösen und auf einen gemeinsamen Nenner zu kommen. Im Interviewverlauf war es den Interviewten gestattet, Verständnisfragen zu stellen und es wurde bei wahrgenommen Unverständnissen bei Antworten erneut nachgefragt. Dies hat im Grundsätzlichen, dass die Ergebnisse eine gute Basis geboten haben, die Forschungsfragen zu beantworten. Dennoch sind bei den Ergebnissen und im Interviewverlauf aufgefallen, dass es trotzdem zu Unverständnissen gekommen ist. So hat z.B. der Interviewte 6 im Interviewverlauf ein mehrdeutiges Verständnis von Tracing, womit Antworten im Detail geäußert wurden, die nicht zum eigentlichen Themenfeld gepasst haben. Zudem hat der Begriff der Sicherheitsrelevanz dazu geführt, dass die Interviewten sich auf das generelle Thema der IT-Sicherheit fokussiert haben und den Fokus auf den Traceability Aspekt verloren haben. Diese Probleme können für spätere Reproduktionen des Interviews aber dadurch beseitigt werden, dass bei abweichenden Antworten der Interviewten aktiv der Interviewte auf die eigentliche Intention der Fragestellung hingewiesen wird.

6.3.2 Interne Validität

Interne Validität zeigt, ob die Veränderungen, die wir in der Arbeit sehen, wirklich durch das verursacht werden, was wir manipuliert haben. Wir wollen sicherstellen, dass keine anderen Faktoren die Ergebnisse stören [92]. Die Interne Validität ist durch die qualitative Natur des Interviewformats negativ beeinträchtigt. Dennoch war es das Ziel der Arbeit, Personen aus der Praxis zu befragen. Damit ist die Grundprämisse, dass theoretische Aspekte des Themas vernachlässigt werden und die praktischen Aspekte, seien sie vollkommen oder fehlerhaft, im Vordergrund stehen. Um vielfältige Sichtweisen aus der Praxis zu bekommen, wurden Interviewten mit verschiedenen Rollen und jeweils aus fünf verschiedenen Unternehmen befragt. Das hat es ermöglicht, dass wir keine Voreingenommenheit in den Ergebnissen durch mehrere Personen als Interviewpartner haben, die dieselben Arbeitsumstände haben. Ein zentrales Problem spezifisch für unser Thema war die Sensibilität des Themas. Dadurch, dass die Offenlegung der Zustände, wie sicherheitsrelevante Artefakte getraced werden, ein potenzielles Sicherheitsrisiko darstellen kann, besteht das Risiko bei diesen und ähnlichen Themen, dass in Interviews Informationen in voller Gänze nicht preisgegeben. Praktisch hat sich dies aber nicht als Problem dargestellt, da offen über sicherheitsrelevante Themen in Detail geredet wurde und interne Praktiken und Methodiken. Zudem wurde offen auch über Probleme geredet, womit selbst über negativen Zustände innerhalb der Firma transparent geredet wurde. Dies ist insbesondere darauf zurückzuführen, da die Interviews anonymisiert werden und in Rohfassung nicht veröffentlicht werden.

6.3.3 Externe Validität

Externe Validität betrachtet, ob wir die Ergebnisse unseres Interviews auf andere Situationen außerhalb unseres Interviewrahmens übertragen können. Es geht darum, dass die Ergebnisse auch anderswo gültig sein sollten [92]. Durch die wie schon vorher erwähnte Unternehmensvielfalt wurden Interviewte tätig in verschiedenen Unternehmensdomänen erfragt, womit sichergestellt ist, dass das Interview nicht beschränkt auf eine bestimmte Unternehmensdomäne ist. Dennoch ist es durch die niedrige Anzahl von acht Interviews nicht möglich, auf den allgemeinen Zustand der Praxis zu schließen. Dies ist vor allem dadurch ersichtlich, dass wir selbst im achten und letzten Interview neue Erkenntnisse in den jeweiligen Fragenkategorien sammeln konnten, besonders in Forschungsfrage 1 (dadurch, dass die Fragestellung offene Antworten erlaubt). Es wurde aber darauf geachtet, dass die Interviewten möglichst viel Berufserfahrung und Kenntnisse über das Thema besitzen. So war es also potenziell möglich, durch mehr Interviewte mehr Erkenntnisse zu sammeln, aber die Qualität an Antworten war durch den Kenntnisstand dennoch gut genug, dass diese zur Beantwortung der Forschungsfragen genügt haben. Auf Basis dieser Erkenntnisse können weitere Arbeiten in Zukunft durchgeführt werden, die dann bestimmte Aspekte im Detail näher erleuchten. Die bereits entwickelten Handlungsempfehlungen berücksichtigen diese Aspekte der externen Validität und sollten auch auf ähnliche Unternehmensdomänen anwendbar sein.

6.3.4 Experimentelle Validität

Reliabilität bedeutet, dass wenn jemand anderes das gleiche Interview unter den gleichen Bedingungen durchführt, er ähnliche Ergebnisse erhalten sollte. Die Ergebnisse sollten zuverlässig und wiederholbar sein [92]. Ein Interviewleitfaden und die Beschreibung, wie das Interview durchgeführt wurde, wurde erstellt und ist im Anhang A vorzufinden. Somit ist es möglich, in der Zukunft das Interview zu reproduzieren. Ähnliche Ergebnisse sind aufgrund der verschiedenen Methodiken und Prozessen innerhalb der Unternehmen schwer möglich sein, auch dadurch, dass das Thema Tracing bei jedem Unternehmen wie auch schon in den Interviews festgestellt unterschiedlich gehandhabt wird. Ähnliche Muster und Feststellungen, insbesondere bei der Wahrnehmung von Tracing und wie es teils durchgeführt wird, sollten sich auch bei zukünftigen Arbeiten vorfinden können. Mehr Interviewte sollten es zudem ermöglichen, mehr Erkenntnisse zu erzielen, da die Erkenntnissättigung noch nicht erreicht wurde. Die Codes wurden rein induktiv ermittelt, waren also basierend darauf, was persönlich relevant war auf Basis des Kenntnisstandes. Selbst innerhalb der Ergebnisse dieser Arbeit können leicht andere Ergebnisse ermittelt werden, da Feinheiten

in den Informationen ermittelt werden können, die mehr Kenntnisse und Erfahrungen erfordern. Die technische Natur des Themas schränkt aber die Interpretationsmöglichkeiten der Ergebnisse stark ein, womit selbst bei unterschiedlichen Personen, die das Interview codieren, zumindest ähnliche Ergebnisse ermittelt werden sollten.

Kapitel 7

Fazit und Ausblick

7.1 Zusammenfassung der Arbeit

Im Rahmen dieser Arbeit wurde untersucht, wie das Tracing von sicherheitsrelevanten Artefakten in der Praxis durchgeführt wird. Die zentrale Herausforderung liegt dabei in der Identifikation und Verfolgung dieser Artefakte, um die Einhaltung von Sicherheitsstandards zu gewährleisten und Sicherheitslücken zu minimieren. Zu diesem Zweck wurde eine qualitative Interviewstudie durchgeführt, bei der acht Interviewte nach dem Stand des Tracings in ihrem Unternehmen befragt wurden. Im ersten Teil des Interviews wurde untersucht, welche Artefakte die Unternehmen als sicherheitsrelevant identifizieren und nach welchen Kriterien und Eigenschaften sie dies tun. Die Ergebnisse zeigen, dass die Definition sicherheitsrelevanter Artefakte in hohem Maße von der jeweiligen Domäne und den spezifischen Phasen des Entwicklungsprozesses abhängt. Sicherheitsstandards spielen eine zentrale Rolle, wobei sowohl branchenspezifische als auch übergreifende Normen wie ISO 27001 zur Anwendung kommen. Die durchgeführten Interviews bestätigen, dass eine frühzeitige Integration der Sicherheitsbetrachtung, entsprechend dem Shift-Left-Ansatz, in den Unternehmen zunehmend verankert ist. Weiterhin zeigt sich ein Unterschied in der Auffassung von Sicherheitsrelevanz als Safety oder Security, da sich hierbei die Art der Artefakte unterscheidet. Im größeren zweiten Fragenteil des Interviews wurde erörtert, wie das Tracing im Detail aussieht. Die interviewten Personen wenden Tracing in unterschiedlichem Umfang an. Die eine Hälfte der Befragten differenziert zwischen sicherheitsrelevanten und nicht-sicherheitsrelevanten Artefakten, während die andere Hälfte alle Artefakte gleich behandelt. Die Unterscheidung erfolgt in Abhängigkeit von der Funktion, den Sicherheitsanforderungen, dem Zeitmangel oder der Sicherheitsvernachlässigung. Wichtige Tools wie Jira, Polarion, Enterprise Architect, IBM Doors und JAMA werden eingesetzt, wobei Jira vor allem für das Projektmanagement und JAMA speziell für zertifizierte domänenspezifi-

sche Sicherheitsstandards wie die ISO 26262 verwendet wird. Tracing wird in den Softwareentwicklungszyklus integriert, allerdings ist die Automatisierung begrenzt und mit Komplexitäts- und Fehleranfälligkeitsproblemen behaftet. Die Einhaltung von Sicherheitsstandards wird durch Compliance und Audits gewährleistet, wodurch die Transparenz und die Einhaltung der vorgeschriebenen Sicherheitsrichtlinien verbessert werden. Die Kommunikation spielt eine entscheidende Rolle im Tracingprozess, wobei Softwarearchitekten und Functional Safety Manager eine Schlüsselrolle einnehmen. Die Mehrheit der Befragten sieht keine spezifischen Nachteile beim Tracing, außer einem Mehraufwand durch Schulung und Implementierung, der nicht direkt mit dem Tracing zusammenhängt. Schwierigkeiten bestehen hauptsächlich in der Verwaltung der Trace Links und des Tooling. Sicherheitsbedenken hinsichtlich der Tracingdaten lassen sich durch gängige Sicherheitsmaßnahmen bewältigen. Die Integration des Tracings in die Prozessplanung sowie der Erwerb entsprechender Kenntnisse stellen dabei wesentliche Faktoren dar. Basierend auf den durchgeführten Interviews wurden Handlungsempfehlungen formuliert, die darauf abzielen, das Bewusstsein für Sicherheitsstandards zu stärken, die Automatisierung von Tracingprozessen zu erhöhen, die Traceability von sicherheitsrelevanten Artefakten verpflichtend zu machen, eine klare Kommunikation und interdisziplinäre Zusammenarbeit zu fördern, Externe in das Tracing miteinzubeziehen und den Zugriff auf sicherheitsrelevante Tracingdaten vor Unbefugten zu schützen.

7.2 Ausblick für weitere Arbeiten

Diese Arbeit bietet einen generellen Überblick über das Forschungsthema, wobei Details nach dem aktuellen Stand der Forschung noch nicht im Detail ausgearbeitet sind. So kann beispielsweise in einer Folgestudie untersucht werden, inwiefern sich die Ergebnisse unterscheiden für Interviewte, die primär in Domänen arbeiten, in denen Security einen Fokus hat, im Kontrast zu Safety. Dies ist deshalb von Interesse, da die Erkenntnisse eine Tendenz hatten, von Safety-Aspekten beeinflusst worden zu sein. Darüber könnte ein Modell erstellt werden, die einen tiefen Einblick in die konkrete Ausgestaltung eines Prozesses für das Tracing von sicherheitsrelevanten Artefakten bietet. Hier könnte eine Erweiterung bestehender Richtlinien und Standards, die Traceability als Prozess definieren, erfolgen, indem Aspekte, die eine besondere Relevanz bei dem Tracing von sicherheitsrelevanten Aspekten haben, in den Prozessfokus rücken. In der vorliegenden Arbeit wurde kein genauerer Vergleich von generischen und sicherheitsrelevanten Artefakten gezogen. Dies ist von Relevanz, um im Näheren zu verstehen, ob bestimmte Probleme und Herausforderungen im Tracing von sicherheitsrelevanten Artefakten bestehen, die jeweils exklusiv sind. Hier könnte in einem Folgeinterview eine Differenzierung in zwei Interviewgruppen

erfolgen, wobei die Methodiken im Näheren untersucht werden. Die Ergebnisse zeigen zudem, dass in der Praxis, zumindest ausgehend von den Interviewten, semiautomatische Tracing-Tools verwendet werden, die jeweils aber automatisierte Vorschläge oder Konfigurationen für Tracelinks vorgeben. Viele der von den Interviewten beschriebenen Probleme sind auf Fehler oder Nachlässigkeiten der Akteure in der Softwareentwicklung zurückzuführen, wodurch eine Automatisierung Abhilfe schaffen könnte. Es wäre von Vorteil, die Gründe für die Nicht-Anwendung bestehender KI-Modelle für das Tracing zu eruieren und Möglichkeiten zur Anpassung dieser Modelle zu ermitteln, um den Belangen der Praktiker gerecht zu werden. Im Kontext des Tracings sicherheitsrelevanter Artefakte ist eine Anpassung der Modelle von besonderer Relevanz, da erhöhte Compliance-Richtlinien das Training und die Anwendung der Modelle erschweren könnten. Zudem sollten potenzielle Anwendungsfälle, die aus dem Tracing sicherheitsrelevanter Artefakte resultieren, näher untersucht werden. In den geführten Interviews wurde vereinzelt darauf hingewiesen, dass die Nachvollziehbarkeit von Artefakten dazu führt, dass das System besser verstanden wird und folglich sicherheitsrelevante Schwachstellen aufgedeckt werden können. Es wäre empfehlenswert, diese sekundäre Eigenschaft gezielt zu nutzen und Methodiken zu konzipieren, um spezifische Risiken und Schwachstellen im System effektiv zu identifizieren und entsprechende Gegenmaßnahmen zu entwickeln.

Anhang A

Anhang

A.1 Einverständniserklärung

Einverständniserklärung zur Teilnahme an der Interviewstudie zum Tracing von sicherheitsrelevanten Artefakten in der Praxis

Diese Studie wird im Rahmen der Masterarbeit von Abdurrahman Sekerci an der Leibniz Universität Hannover am Fachgebiet Software Engineering mit dem Titel „Interviewstudie zum Tracing von sicherheitsrelevanten Artefakten in der Praxis“ durchgeführt. Eine Teilnahme an der Interviewstudie ermöglicht es zu erforschen, wie sicherheitsrelevante Artefakte in der Softwareentwicklung definiert werden. Zudem wird untersucht, wie das Tracing der sicherheitsrelevanten Artefakte aussieht, mit welchem Methoden und Prozessen das Tracing geschieht und welche Probleme und Herausforderungen bestehen.

Das Interview wird aufgezeichnet und im Anschluss zur Analyse verschriftlicht. Dabei werden die Daten anonymisiert und es wird kein Rückschluss auf Ihre Person möglich sein. Die Daten werden im Rahmen der Masterarbeit ausgewertet und präsentiert. Darüber hinaus können sie in anderen wissenschaftlichen Arbeiten anonymisiert veröffentlicht werden. Die anonymisierten Aussagen können dabei anonymisiert veröffentlicht werden. Es werden keine Rohdaten veröffentlicht.

Die erfassten Daten werden elektronisch unbegrenzt gespeichert. Nach Abschluss der Masterarbeit werden die Daten auf institutseigenen Servern liegen. Ein Zugriff durch externe Personen ist zu keinem Zeitpunkt möglich.

Sie nehmen freiwillig und ohne Vergütung an dieser Studie teil. Sie haben das Recht, die Teilnahme jederzeit und ohne Angabe von Gründen abzubrechen, ohne dass dies zu negativen Konsequenzen für Sie führt.

Gemäß dem Datenschutz gegenüber dem Informationsträger haben Sie das Recht auf Auskunft sowie Löschung Ihrer personenbezogenen Daten. Diese Einverständniserklärung kann jederzeit widerrufen werden. Nach erfolgtem Widerruf werden ihre personenbezogenen Daten gelöscht und für keine Publikation mehr verwendet.

Mit den aufgeführten Punkten bin ich

☐ einverstanden.

☐ nicht einverstanden.

Nachname, Vorname: _____

Ort, Datum, Unterschrift: _____

A.2 Interviewleitfaden

Beginn des Interviews – Einleitung

Zur Vereinfachung des Interviewverlaufs wird in der schriftlichen Fassung des Leitfadens durchgehend gesiezt. In der mündlichen Ausführung ist Duzen dennoch gestattet.

- Begrüßung des Teilnehmers und Danksagung für die Teilhabe am Interview
- Audioaufnahme starten
- Das Einverständnis zur Teilnahme erneut mündlich sicherstellen, mit der Bedingung, dass das Interview aufgezeichnet wird
- Persönliche Vorstellung (falls Teilnehmer unbekannt)
  - *Ich bin Abdurrahman Sekerci, studiere an der Leibniz Universität Hannover im Fach Informatik mit dem Abschlussgrad des Masters*
- Nennung des Abschlussarbeitsnamens
  - *Ich bin hier, um eine Interviewstudie mit Ihnen/Dir für meine Masterarbeit mit dem Titel „Interviewstudie zum Tracing von sicherheitsrelevanten Artefakten in der Praxis“ durchzuführen*
- Vorstellung des Themas
  - Erkenntnisse sammeln, wie das Tracing von sicherheitsrelevanten Artefakten in der Industrie aussieht, um im Nachhinein Handlungsempfehlungen/Modell zu entwerfen
  - **Dem Interviewteilnehmer gerichtet**: Nachfragen wie Tracing oder sicherheitsrelevante Artefakte verstanden werden und bei Unverständnissen aufklären und Definition geben
- Vorstellung des Ablaufs
  - Soziodemografische Fragen zu der Person des Intervierpartners
  - Fragen zur Sicherheitsrelevanz von Artefakten
  - Fragen zum Tracing und spezifisch für das Tracing von sicherheitsrelevanten Artefakten
  - Dem Interviewpartner vergewissern, dass das Interview aufgezeichnet wird, danach transkribiert und anonymisiert wird für den Datenschutz
  - Dem Interviewpartner mitteilen, dass das Interview freiwillig ist und er jederzeit das Recht hat, deswegen das Interview abzubrechen

  - Die Nichtbeantwortung der Frage ist möglich und das Interview wird einfach weitergeführt

Soziodemografische Fragen (Antwortmöglichkeiten in Form von kurzen oder geschlossenen Antworten)

- Vor dem Stellen der soziodemografischen Fragen wird der Interviewteilnehmer gebeten, seine Antworten möglichst knappzuhalten, da Interviewzeit beschränkt
- *Wie lange arbeiten/arbeitest Sie/du in der Softwareentwicklung?* (Nur Jahreszahl angeben als Antwort)
- *Wie lange arbeiten/arbeitest Sie/du in Ihrer/deiner aktuellen Rolle im Unternehmen?* (Nur Jahreszahl angeben als Antwort)
- *Was ist deine aktuelle Rolle in dem Unternehmen?* (Nur Rolle angeben als Antwort)

Die folgenden Fragen erlauben offene Antworten
Forschungsfrage 1: Was sind sicherheitsrelevante Artefakte in der Praxis?

- *Können sie Beispiele für sicherheitsrelevante Artefakte in ihren Unternehmen nennen?*
- *(Optional): Inwiefern tragen diese Artefakte zur Sicherheit der Software bei?*
- *Wie identifizieren Sie diese Artefakte in Ihren Projekten?*
  - **(Wenn Identifizierung nicht vorhanden):** *Gibt es spezielle Gründe, warum darauf verzichtet wird?*
  - **(Wenn Identifizierung vorhanden):** *Werden diese Artefakte im Generellen unterschiedlich behandelt?*
- **(Wenn Identifizierung vorhanden)**: *Können Sie beschreiben, anhand welcher Eigenschaften Artefakte in Ihrem Team oder Unternehmen als sicherheitsrelevant betrachtet werden?*
- **(Wenn Identifizierung nicht vorhanden)**: *Können Sie beschreiben, anhand welcher Eigenschaften Artefakte aus ihrer persönlichen Einschätzung als sicherheitsrelevant betrachtet werden können?*

- **Optional zu Eigenschaften**: *Welche Prozesse oder Richtlinien bestimmen die Bestimmung der Sicherheitsrelevanz?*

Forschungsfrage 2.1: Wird Tracing für diese Artefakte betrieben?

- Wird in Ihrem Unternehmen Tracing für sicherheitsrelevante Artefakte betrieben?
  - **(Wenn nein):** *Gibt es spezielle Gründe, warum darauf verzichtet wird?*
  - **(Wenn ja):** *Gibt es bei sicherheitsrelevanten Artefakten einen Unterschied zum Tracing von generischen Artefakten? Warum oder warum nicht?* (Hierbei dem Interviewpartner betonen, dass die Antwort möglichst kurz sein soll und sich auf das Warum fokussiert wird)
- Können sie Beispiele für Trace Links zwischen sicherheitsrelevanten Artefakten nennen?
- Welche Vorteile sehen Sie durch das Tracing?
- **(Wenn keine Anwendungsfälle genannt werden:)** *Können sie mir konkrete Anwendungsbeispiele nennen, wo das Tracing der sicherheitsrelevanten Artefakte ihnen geholfen hat?*

Abhängig von der Antwort zur vorherigen Frage, ob es hierbei spezifische Unterscheide zum generischen Tracing gibt, werden die darauffolgenden (allgemein gehaltenen) Fragen anders gestellt. Wenn es Unterschiede gibt, muss der Interviewpartner darauf hingewiesen werden, dass er oder sie in den Teilaspekten auch die Unterschiede betont und möglichst sich auf diese konzentriert. Wenn es keine Unterschiede gibt, soll der Interviewpartner möglichst gut hervorheben, wie und warum der Teilaspekt bei sicherheitsrelevanten Artefakten angepasst werden könnte (basierend auf den Erfahrungen in der Softwareentwicklung). Es können zusätzlich auch Erfahrungswerte aus älteren Unternehmen genannt werden.

Forschungsfrage 2.2: Mit welchen Methoden und Prozessen wird dieses Tracing durchgeführt?

- *Welche spezifischen Traceability-Tools oder Softwarelösungen setzen Sie für die Nachverfolgung von sicherheitsrelevanten Artefakten ein?*

- ***(Wenn das Tool im Näheren nicht erklärt wird:)*** *In welchem Kontext und zu welchem Zweck werden diese Tools speziell für sicherheitsrelevante Artefakte verwendet?*
- *Welche Rolle spielt die Traceability von sicherheitsrelevanten Artefakten bei der Einhaltung von Sicherheitsstandards und Compliance-Anforderungen?*
- **(Wenn keine Beispiele zu Compliance Aspekten genannt wird:)** *Können Sie Beispiele nennen, bei dem das effektive Tracing von sicherheitsrelevanten Artefakten geholfen hat, Compliance-Herausforderungen zu meistern oder regulatorische Hürden zu überwinden?*
- *Welche Rolle spielen Sicherheitsaudits und -bewertungen im Kontext des Tracings von sicherheitsrelevanten Artefakten?*
- *Wie wird das Tracing von sicherheitsrelevanten Artefakten in Ihre bestehenden Entwicklungs- und Qualitätssicherungsprozesse integriert?*
- **(Wenn nicht genug Bezüge zu Sicherheitsprozessen erstellt wird):** *Wie ist das Tracing von sicherheitsrelevanten Artefakten in den gesamten Sicherheitsentwicklungszyklus eingebettet? Welche Vorteile erfahren sie dadurch in Bezug auf die Sicherheit der Software?*
- *Gibt es spezielle Anpassungen oder Herausforderungen bei der Integration von Tracing für sicherheitsrelevante Artefakte in diese Prozesse?*
- *Inwieweit ist das Tracing von sicherheitsrelevanten Artefakten in Ihrem Unternehmen automatisiert?*
- **(Wenn Tracing gar nicht oder teilweise nicht betrieben wird:)** *Gibt es spezielle Gründe, warum nicht automatisiert wird?*
- *Welche Vor- und Nachteile sehen Sie in der Automatisierung des Tracing-Prozesses für sicherheitsrelevante Artefakte?*
- *Wie wird die Zusammenarbeit und Kommunikation zwischen den beteiligten Personen im Tracing-Prozess von sicherheitsrelevanten Artefakten gefördert?*
- **(Wenn nicht genug Beispiele für gelungene soziale Phänomenen genannt wird:)** *Können Sie Best Practices oder spezifische Fallbeispiele nennen, in denen die Zusammenarbeit und Kommunikation im Tracing-Prozess besonders effektiv waren?*
- *Gibt es rollenspezifische Unterschiede beim Tracing von sicherheitsrelevanten Artefakten?*

Forschungsfrage 2.3: Welche Probleme bestehen bei diesem Tracing?

- *Sehen sie konkret Nachteile durch Tracing?*
- *Welche Herausforderungen oder Probleme treten bei der Implementierung und Durchführung von Tracing auf?*
- **(Wenn Antwort nicht konkret genug bezüglich Sicherheit ist):** *Welche spezifischen Sicherheitsrisiken und Herausforderungen müssen bei der Nachverfolgung von sicherheitsrelevanten Artefakten berücksichtigt werden?*
- *Wie gehen Sie mit diesen Herausforderungen um?*

Schluss

- Fragen, ob der Teilnehmer zum Interview noch etwas inhaltlich ergänzen kann (weitere Aspekte, Antwort vergessen bei vorherigen Fragen)
- Für die Teilnahme danken
- Audioaufnahme beenden
- Verabschiedung des Teilnehmers

Literaturverzeichnis

[1] A. Agrawal and J. Cleland-Huang. Leveraging traceability to integrate safety analysis artifacts into the software development process. In *2023 IEEE 31st International Requirements Engineering Conference Workshops (REW)*, pages 475–478, 2023.

[2] M. Ahrens and L. Nagel. All eyes on traceability: An interview study on industry practices and eye tracking potential. In *2023 IEEE 31st International Requirements Engineering Conference (RE)*, pages 77–88, 2023.

[3] C. Alberts, A. Dorofee, J. Stevens, and C. Woody. Introduction to the octave approach, 2003.

[4] N. Ali, Z. Sharafl, Y.-G. Guéhéneuc, and G. Antoniol. An empirical study on requirements traceability using eye-tracking. In *2012 28th IEEE International Conference on Software Maintenance (ICSM)*, pages 191–200, 2012.

[5] Amazon Web Services. Amazon Transcribe, 2024. `https://aws.amazon.com/transcribe/` letzter Zugriff am 30. April 2024.

[6] P. Arkley and S. Riddle. Overcoming the traceability benefit problem. In *13th IEEE International Conference on Requirements Engineering (RE'05)*, pages 385–389, 2005.

[7] H. Assal and S. Chiasson. Security in the software development lifecycle. In *Fourteenth Symposium on Usable Privacy and Security (SOUPS 2018)*, pages 281–296. USENIX Association, 2018.

[8] H. U. Asuncion, A. U. Asuncion, and R. N. Taylor. Software traceability with topic modeling. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering - Volume 1*, ICSE '10, page 95–104, New York, NY, USA, 2010. Association for Computing Machinery.

[9] T. Aung, H. Huo, and Y. Sui. Interactive traceability links visualization using hierarchical trace map. In *2019 IEEE International Conference on*

Software Maintenance and Evolution (ICSME), pages 367–369. IEEE, 2019.

[10] T. Aven. *Foundations of Risk Analysis*. John Wiley & Sons, second edition edition, 2012.

[11] M. Bain. WhisperX: Automatic Speech Recognition with Word-level Timestamps (& Diarization). `https://github.com/m-bain/whisperX`, 2022. Letzter Zugriff am 3. März 2024.

[12] L. Braz and A. Bacchelli. Software security during modern code review: The developer's perspective. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 810–821. ACM, 2022.

[13] Bundesamt für Sicherheit in der Informationstechnik. Kritische Log4ShellSSchwachstelle in weit verbreiteter Protokollierungsbibliothek Log4j (CVE-2021-44228). `https://www.bsi.bund.de/SharedDocs/Cybersicherheitswarnungen/DE/2021/2021-549177-1032.pdf?__blob=publicationFile&v=7#download=1`, 2021. Letzter Zugriff am 21. Januar 2024.

[14] Bundesamt für Sicherheit in der Informationstechnik. Die Lage der IT-Sicherheit in Deutschland 2023. `https://www.bsi.bund.de/DE/Service-Navi/Publikationen/Lagebericht/lagebericht_node.html`, 2023. Letzter Zugriff am 17. Januar 2024.

[15] Bundesministerium der Justiz und für Verbraucherschutz. Datenschutz-Grundverordnung. `https://www.bmj.de/DE/themen/digitales/DSGVO/DSGVO_node.html`, 2023. Letzter Zugriff am 20. Dezember 2023.

[16] A. Burns and R. I. Davis. Mixed criticality systems - a review, Feb 2022. 13th Edition.

[17] E. B.V. Account lockout. `https://www.sciencedirect.com/topics/computer-science/account-lockout`, 2021. Letzter Zugriff am 20. Dezember 2023.

[18] J. Cleland-Huang, O. C. Z. Gotel, J. Huffman Hayes, P. Mäder, and A. Zisman. Software traceability: Trends and future directions. In *Future of Software Engineering Proceedings*, FOSE 2014, page 55–69, New York, NY, USA, 2014. Association for Computing Machinery.

[19] G. community. Git. `https://git-scm.com/`. letzter Zugriff am 10.Dezember 2023.

[20] J. W. Creswell. *Qualitative inquiry and research design: Choosing among five traditions*. Sage Publications, 1998.

[21] A. De Lucia, A. Marcus, R. Oliveto, and D. Poshyvanyk. Information retrieval methods for automated traceability recovery. In J. Cleland-Huang, O. Gotel, and A. Zisman, editors, *Software and Systems Traceability*, pages 71–98. Springer London, London, 2012.

[22] R. Debouk and J. Joyce. Iso 26262 hazard and risk assessment methodology. In *International System Safety Conference*, 08 2010.

[23] P. S. Dipl.-Ing. (FH) Stefan Luber. Was ist ein Pufferüberlauf (Buffer Overflow)? *Security-Insider*, Jun 2021. letzter Zugriff am 04. Juni 2024.

[24] A. Egyed, P. Grünbacher, M. Heindl, and S. Biffl. Value-based requirements traceability: Lessons learned. In *Design Requirements Engineering: A Ten-Year Perspective*, volume 14 of *Lecture Notes in Business Information Processing*, pages 240–257. Springer, 2009.

[25] Ericsson. Codechecker. `https://marketplace.visualstudio.com/items?itemName=codechecker.vscode-codechecker`. Letzter Zugriff am 23.Dezember 2023.

[26] European Commission. Radio Equipment Directive (RED). `https://single-market-economy.ec.europa.eu/sectors/electrical-and-electronic-engineering-industries-eei/radio-equipment-directive-red_en`, 2024. Letzter Zugriff am 13. Mai 2024.

[27] R. d. E. U. Europäisches Parlament. Verordnung (EU) 2016/679 des Europäischen Parlaments und des Rates. EUR-Lex, 2016. `https://eur-lex.europa.eu/legal-content/DE/TXT/?uri=CELEX%3A32016R0679` letzter Zugriff am 13. Mai 2024.

[28] M. Eyl, C. Reichmann, and K. Müller-Glaser. Traceability in a fine grained software configuration management system. In *Software Quality: Complexity and Challenges of Software Engineering in Emerging Technologies : 9th International Conference, SWQD 2017, Vienna, Austria, January 17-20, 2017. Ed.: D. Winkler*, volume 269 of *Lecture Notes in Business Information Processing*, pages 15–29. Springer-Verlag, 2017.

[29] D. G. Firesmith. Engineering security requirements. *Journal of Object Technology*, 2(1):53–68, 2003.

[30] J. Fisher, D. Koning, and A. P. Ludwigsen. Utilizing atlassian jira for large-scale software development management. In *ICALEPCS 2013*. Lawrence Livermore National Lab. (LLNL), Livermore, CA (United States), 2013.

[31] I. O. for Standardization. Road vehicles – functional safety – part 3: Concept phase. ISO Standard, 2011.

[32] D. G. für Qualität e.V. *FMEA – Fehlermöglichkeits- und Einflussanalyse*. Beuth Verlag GmbH, Berlin, Wien, Zürich, 5 edition, 2012.

[33] B. G. Glaser and A. L. Strauss. *Discovery of Grounded Theory: Strategies for Qualitative Research*. Aldine Publishing Company, 1967.

[34] V. S. C. S. GmbH. MAXQDA, software for qualitative data analysis. `https://www.maxqda.com`, 2024. letzter Zugriff am 04. Juni 2024.

[35] O. Gotel, J. Cleland-Huang, J. Hayes, A. Zisman, A. Egyed, P. Grünbacher, A. Dekhtyar, G. Antoniol, and J. Maletic. *The Grand Challenge of Traceability (v1.0)*, pages 343–409. Springer-Verlag London Limited, 10 2012.

[36] O. Gotel and C. Finkelstein. An analysis of the requirements traceability problem. In *Proceedings of IEEE International Conference on Requirements Engineering*, pages 94–101, 1994.

[37] J. Guo, J. Cheng, and J. Cleland-Huang. Semantically enhanced software traceability using deep learning techniques. *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*, 2017.

[38] D. Guégan. A note on the interpretability of machine learning algorithms. *SSRN Electronic Journal*, 2020.

[39] J. Hayes, A. Dekhtyar, and S. Sundaram. Advancing candidate link generation for requirements tracing: the study of methods. *IEEE Transactions on Software Engineering*, 32(1):4–19, 2006.

[40] J. H. Hayes and A. Dekhtyar. Humans in the traceability loop: can't live with 'em, can't live without 'em. In *TEFSE '05: Proceedings of the 3rd international workshop on Traceability in emerging forms of software engineering*, TEFSE '05, page 20–23, New York, NY, USA, 2005. Association for Computing Machinery.

[41] R. Hoda, J. Noble, and S. Marshall. Developing a grounded theory to explain the practices of self-organizing agile teams. *Empirical Software Engineering*, 17(6):609–639, 2012.

[42] IBM. Übersicht über DOORS. `https://www.ibm.com/docs/de/engineering-lifecycle-management-suite/doors/9.7.2?topic=engineering-requirements-management-doors-overview`. Letzter Zugriff am 13. Dezember 2023.

[43] IDAP Group. SDLC Design Phase: Definition, Activities, Goals. `https://idapgroup.com/blog/sdlc-design-phase/`, 2022. letzter Zugriff am 18.Dezember 2023.

[44] Institute of Electrical and Electronics Engineers. *IEEE Standard Glossary of Software Engineering Terminology*, 1990. IEEE Std 610-1990.

[45] Jama Software. ISO 26262 und aktuelle Updates: Sicherstellung der funktionalen Sicherheit in der Automobilindustrie. https://www.jamasoftware.com/requirements-management-guide/automotive-engineering/iso-26262-and-recent-updates-ensuring-functional-safety-in-the-automotive-industry, 2024. Letzter Zugriff am 30. April 2024.

[46] J. Jarzombek and K. Goertzel. Security in the software life cycle. *The Journal of Defense Software Engineering*, 19:4–9, 01 2006.

[47] B. Jiang, P. Liu, and J. Xu. A deep learning approach to locate buggy files. In *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, pages 219–223, 2020.

[48] B. Johnson, Y. Song, E. Murphy-Hill, and R. Bowdidge. Why don't software developers use static analysis tools to find bugs? In *2013 35th International Conference on Software Engineering (ICSE)*, page 672–681. IEEE, 2013.

[49] J. Jürjens. *Secure Systems Development with UML*. Springer, 2005.

[50] M. Kersten et al. Eclipse Mylyn Open Source Project | The Eclipse Foundation, 2023. Letzter Zugriff am 27.Dezember 2023.

[51] R. A. Khan, S. U. Khan, H. U. Khan, and M. Ilyas. Systematic mapping study on security approaches in secure software engineering. *IEEE Access*, 9:19139–19160, 2021.

[52] C. Konstantinidou, W. Lang, A. Papadopoulos, and M. Santamouris. Life cycle and life cycle cost implications of integrated phase change materials in office buildings. *International Journal of Energy Research*, 43, 10 2018.

[53] M. B. Line, O. Nordland, L. Røstad, and I. A. Tøndel. Safety vs security? In *PSAM Conference, New Orleans, USA*. sn, 2006.

[54] B. LLC. Git Integration for Jira (GitHub, GitLab and more) | Atlassian Marketplace, 2023. Letzter Zugriff am 27.Dezember 2023.

[55] P. Mäder and A. Egyed. Do developers benefit from requirements traceability when evolving and maintaining a software system? *Empirical Software Engineering*, 20(2):413–441, 2015.

[56] A. Mahmoud, N. Niu, and S. Xu. A semantic relatedness approach for traceability link recovery. In *2012 20th IEEE International Conference on Program Comprehension (ICPC)*, pages 183–192, 2012.

[57] MAXQDA. MAXQDA Transcription, 2024. `https://www.maxqda.com/` letzter Zugriff am 30. April 2024.

[58] P. Mayring and T. Fenzl. Qualitative Inhaltsanalyse. In N. Baur and J. Blasius, editors, *Methoden der empirischen Sozialforschung*, pages 601–613. Springer VS, 3 edition, 2022.

[59] P. H. Meland and J. Jensen. Secure software design in practice. In *2008 Third International Conference on Availability, Reliability and Security*, pages 1164–1171, 2008.

[60] Microsoft. Was ist SIEM?, 2024. `https://www.microsoft.com/de-de/security/business/security-101/what-is-siem` letzter Zugriff am: 13.Mai 2024.

[61] C. Mills, J. Escobar-Avila, and S. Haiduc. Automatic traceability maintenance via machine learning classification. In *2018 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 369–380, 2018.

[62] A. P. Moore, R. J. Ellison, and R. C. Linger. Attack modeling for information security and survivability. Technical Report CMU/SEI-2001-TN-001, Software Engineering Institute, Carnegie Mellon University, 2001. letzter Zugriff am 04. Juni 2024.

[63] Motor Industry Software Reliability Association. MISRA-C:2004 - Guidelines for the use of the C language in critical systems. Motor Industry Research Association, 2004. `www.misra.org.uk)` letzter Zugriff am 04. Juni 2024.

[64] E. Y. Nakagawa and P. O. Antonino. *Reference Architectures for Critical Domains: Industrial Uses and Impacts.* Springer International Publishing, 2023. `https://link.springer.com/book/10.1007/978-3-031-16957-1` letzter Zugriff am 04. Juni 2024.

[65] N. Nazir and M. K. Nazir. A review of security issues in sdlc. *American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)*, 46(1):247–259, 2018.

[66] A. Nhlabatsi, Y. Yu, A. Zisman, T. Tun, N. Khan, A. Bandara, K. M. Khan, and B. Nuseibeh. Managing security control assumptions using causal traceability. In *2015 IEEE/ACM 8th International Symposium on Software and Systems Traceability*, pages 43–49, 2015.

[67] M. Niazi, A. M. Saeed, M. Alshayeb, S. Mahmood, and S. Zafar. A maturity model for secure requirements engineering. *Computers & Security*, 95:101852, 2020. `https://www.sciencedirect.com/science/article/pii/S0167404820301243` Letzter Zugriff am 04. Juni 2024.

[68] OpenAI. Whisper: Robust Speech Recognition via Large-Scale Weak Supervision. `https://github.com/openai/whisper`, 2021. Letzter Zugriff am 3.März 2024.

[69] Otter.ai. Otter.ai: Transcription Service, 2024. `https://otter.ai` letzter Zugriff am 30. April 2024.

[70] Z. Pauzi and A. Capiluppi. Applications of natural language processing in software traceability: A systematic mapping study. *Journal of Systems and Software*, 198:111616, 2023. `https://www.sciencedirect.com/science/article/pii/S0164121223000110` letzter Zugriff am 04. Juni 2024.

[71] N. Pecka, L. ben Othmane, and A. Valani. Making secure software insecure without changing its code: The possibilities and impacts of attacks on the devops pipeline, 2022.

[72] Prof. Dr. rer. nat. Kurt Schneider. Grundlagen der Softwaretechnik Vorlesung Leibniz Universität Hannover Kapitel 1 Seite 16, 2023.

[73] proofpoint. Was ist HIPAA? `https://www.proofpoint.com/de/threat-reference/hipaa-compliance`. Letzter Zugriff: 12.Dezember 2023.

[74] A. Radford et al. Robust speech recognition via large-scale weak supervision. *OpenAI*, 2022.

[75] M. D. Roshaidie, W. P. H. Liang, C. G. K. Jun, K. H. Yew, and F. tuz Zahra. Importance of secure software development processes and tools for developers, 2020.

[76] J. Rowley. Conducting research interviews. *Management Research Review*, 35(3/4):260–271, 2012.

[77] SAFECode. *Fundamental Practices for Secure Software Development: Essential Elements of a Secure Development Lifecycle Program.* Software Assurance Forum for Excellence in Code, 2018.

https://safecode.org/wp-content/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Development_March_2018.pdf Letzter Zugriff am 04. Juni 2024.

[78] J. Saltzer and M. Schroeder. The protection of information in computer systems. *Proceedings of the IEEE*, 63(9):1278–1308, 1975.

[79] SANS Institute. Top 25 software errors. https://www.sans.org/top25-software-errors/, 2023. Letzter Zugriff am 24.Dezember 2023.

[80] C. B. Seaman. Qualitative methods. In *Guide to Advanced Empirical Software Engineering*, pages 35–62. Springer, 2008.

[81] Siemens. Polarion - Software - Application Lifecycle Management (ALM). https://polarion.plm.automation.siemens.com/, 2024. letzter Zugriff am 13. Mai 2024.

[82] G. Spanoudakis and A. Zisman. Software traceability: A roadmap. *Handbook of Software Engineering and Knowledge Engineering*, 3, 08 2005.

[83] M. Stevens, A. K. Lenstra, and B. de Weger. Colliding x.509 certificates for different identities. https://www.win.tue.nl/hashclash/TargetCollidingCertificates/, 2007. Letzter Zugriff am 04. Juni 2024.

[84] A. Strauss and J. Corbin. Grounded theory methodology: An overview. In N. K. Denzin and Y. S. Lincoln, editors, *Handbook of qualitative research*, pages 273–285. Sage Publications, Inc, 1994.

[85] t2informatik. Traceability Matrix. https://t2informatik.de/wissen-kompakt/traceability-matrix/. letzter Zugriff am 05.Dezember 2023.

[86] F. Tian, T. Wang, P. Liang, C. Wang, A. A. Khan, and M. A. Babar. The impact of traceability on software maintenance and evolution: A mapping study, 2021.

[87] C. U. Addressing the lack of software development lifecycle security. https://itsecuritywire.com/featured/addressing-the-lack-of-software-development-lifecycle-security/. Letzter Zugriff am 17. Dezember 2023.

[88] A. I. Wasserman. Tool integration in software engineering environments. In F. Long, editor, *Software Engineering Environments*, pages 137–149, Berlin, Heidelberg, 1990. Springer Berlin Heidelberg.

[89] C. Weir, S. Migues, and L. Williams. Exploring the shift in security responsibility. *IEEE Security & Privacy*, 20(6):8–17, 2022.

[90] E. Weis. Vertraulichkeit, Integrität und Verfügbarkeit - Schutzziele der Informationssicherheit. *BRANDMAUER IT-Security Blog*, 2022. `https://www.brandmauer.de/blog/it-security/schutzziele-der-informationssicherheit` letzter Zugriff am 18.Dezember 2023.

[91] Wikipedia Community. ISO/IEC 27001. `https://de.wikipedia.org/wiki/ISO/IEC_27001`. Letzter Zugriff: 12.Dezember 2023.

[92] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering*. Springer, 2012.

[93] W. Xiong and R. Lagerström. Threat modeling–a systematic literature review. *Computers & Security*, 84:53–69, 2019.

[94] W. Yu and S. Smith. Reusability of fea software: A program family approach. *Proceedings of the 2009 ICSE Workshop on Software Engineering for Computational Science and Engineering, SECSE 2009*, pages 43–50, 05 2009.

[95] H. A. Çetin and E. Tüzün. Analyzing developer contributions using artifact traceability graphs. *Empirical Software Engineering*, 27(3):77, 2022. `https://doi.org/10.1007/s10664-022-10129-2` letzter Zugriff am 04. Juni 2024.