Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

DELIA Abschlussbericht

Prof. Dr. Mathias Fischer, Doğanalp Ergenç
Rechnernetze (NET)
Fachbereich Informatik
Universität Hamburg

31. Januar 2023

Förderkennzeichen:	16KIS0941
Fördergeber:	BMBF
Projektkoordinator:	ZAL GmbH
Konsortium:	ZAL GmbH, AED Engineering GmbH, Solectrix GmbH, Universität Stuttgart, Universität Hamburg, Airbus Operations (assoziiert), Tyco Electronics Raychem GmbH (assoziiert)

Inhaltsverzeichnis

1 Kurzbeschreibung des Projekts

1.1 Allgemeine Ziele

Moderne mission-critical systems (MCS), z. B. wie sie in der Automobil- und Avionikbranche zu finden sind, sind heterogene und komplexe Ökosysteme. Sie bestehen aus einer Vielzahl von miteinander verbundenen Komponenten und internen Funktionen und zeichnen sich durch eine große Vielfalt unterschiedlicher Hardware und Software aus. Diese alles erschwert die Entwicklung solcher Systeme und schränkt ihre Erweiterbarkeit und Rekonfigurierbarkeit ein. Daher werden mittlerweile verstärkt neue Technologien, wie Virtualisierung, Service-oriented Architectures (SOA) oder Time-sensitive Networking (TSN) in MCS angewendet, um ihre Entwicklung zu erleichtern. Die Virtualisierung eingebetteter Komponenten ermöglicht die Anwendung von SOA für sicherheitskritische Systeme. Kritische Systemdienste können so flexibel und virtualisiert auf General-Purpose-Hardware realisiert werden, anstatt auf dedizierter Hardware mit begrenzten Fähigkeiten zu laufen. In ähnlicher Weise helfen die jüngsten IEEE-Standards für zeitkritische Netzwerke (TSN) dabei, deterministische Kommunikation zwischen gemischt-kritischen Diensten und mittels handelsüblicher und daher günstiger Hardware zu realisieren. Diese neuen Technologien ermöglichen zudem die Verwendung neuer Methoden für die Erhöhung von Sicherheit und Zuverlässigkeit von MCS insbesondere auch gegenüber Fehlern und Cyber-Angriffen.
Das Projekt DELIA und insbesondere auch das Teilprojekt der Universität Hamburg zielte darauf ab, eine skalierbare, leichtgewichtige, erweiterbare und resiliente Kommunikations- und Verarbeitungsplattform auf der Basis von SOA und TSN zu entwickeln. Die UHH hat sich dazu vor allem auf die Entwicklung der erforderlichen Software-Module fokussiert die auf DELIA-Hardware-Modulen lauffähig sind. DELIA-Instanzen sollen in der Lage sein, kritische und nicht kritische virtuelle Dienste isoliert zu hosten und miteinander zu kommunizieren, um komplexe Aufgaben kritischer Netze zu bewältigen, wobei der Schwerpunkt auf Avioniksystemen liegt.

1.2 Projektstruktur

Im Hinblick auf die vorgegebenen Gesamtziele bestand die Hauptaufgabe der UHH darin, sichere System-Software-Funktionen zu entwickeln sowie Sicherheits- und Zuverlässigkeitsmaßnahmen sowohl für die Verarbeitung als auch für die Kommunikation der DELIA-Knoten vorzuschlagen. Der Rest dieses Abschnitts beschreibt kurz alle Arbeitspakte, an denen wir beteiligt waren.

HAP1: Anforderungensentwicklung

Für das HAP1 sollten die Anforderungen an die Gestaltung der dienstorientierten Architektur und der zeitkritischen Kommunikation definiert werden. Die UHH war insbesondere für die Entwicklung der Systemarchitektur mit einem starken Fokus auf die Sicherheit der Dienstverteilung und der dienstübergreifenden Kommunikation verantwortlich. Dazu gehörten die Definition der Sicherheitsanforderungen an die Verarbeitungs- und Kommunikationsplattform, die Identifikiation potenzieller Bedrohungen, die Erstellung von Angreifermodellen sowie die Analyse der vorhandenen Software und die Adaption bewährter Sicherheitspraktiken, z.B. zur Härtung des Betriebssystems von DELIA-Knoten. Dementsprechend haben wir uns an den folgenden Arbeitspaketen beteiligt:

AP1.2: Funktionsverteilung & Sicherheitsanforderungen
UHHs Rolle: AP 1.2.UHH.1 Sicherheitsanforderungen, AP 1.2.UHH.2 Angreifermodell

AP1.3: Systemarchitektur
UHHs Rolle: AP 1.3.UHH.1 Sichere Softwarearchitektur

HAP2: Domänenspezifisches Entwurf und Realisierung

Das HAP2 befasst sich mit der Konzeption und Entwicklung von Hardware- und Softwaremodulen für die DELIA-Knoten. Die UHH war hauptverantwortlich für die Analyse und Entwicklung der Systemsoftware, die die Verteilung und Verwaltung von Diensten ermöglicht und Sicherheitsfunktionen für das Gesamtsystem einsetzt. Dazu gehört auch die Implementierung der Plattform-Firmware und von Monitoring-Tools für die Überwachung der Plattform auf Fehler und Angriffe und für die Wartung. Über das Design der einzelnen DELIA-Knoten hinaus haben wir auch am Aufbau einer resilienten Kommunikationsplattform mitgewirkt, d. h. einer sicheren, fehlertoleranten Vernetzung zwischen den DELIA-Knoten. Dementsprechend war die UHH an den folgenden Arbeitspaketen beteiligt:

AP2.1: Plattform Hardware und Firmware
UHHs Rolle: AP 2.1.UHH.1 Sichere Plattformfirmware

AP2.2: Betriebssysteme & Applikationen
UHHs Rolle: AP 2.2.UHH.1 Sichere Plattform und Systemfunktionen,
AP 2.2.UHH.2 Plattformsensor

AP2.3: Debug- und Diagnosetools
UHHs Rolle: AP 2.3.UHH.1 Plattformüberwachung, AP 2.3.UHH.2 Angriffserkennung,
AP 2.3.UHH.3 Fehlersuch- und Forensikwerkzeuge, AP 2.3.UHH.4 Automatische Reparatur

HAP3: Demonstration und Erprobung

Für das HAP3 schließlich sollte das Konsortium seine einzelnen Beiträge in das DELIA-Gesamtsystem integrieren, das als gemeinsamer Demonstrator gebaut wird, um die ausgewählten avionikspezifischen Szenarien zu zeigen. Die Hauptaufgabe der UHH bestand darin, den entwickelten Software-Stack auf der DELIA-Hardware einzusetzen, Sicherheits- und Zuverlässigkeitsmechanismen zu integrieren und deren Wirksamkeit gegen Sicherheitsbedrohungen zu demonstrieren. Wir waren an den folgenden Arbeitspaketen beteiligt:

AP3.1: Systemintegration
UHHs Rolle: AP 3.1.UHH.1 Integration der Sicherheitslösungen

AP3.2: Aufbau eines Verbunddemonstrators
UHHs Rolle: AP 3.2.UHH.1 Sicherer Demonstrator

Der nächste Abschnitt fasst unsere allgemeinen Beiträge zu den genannten Arbeitspaketen zusammen.

1.3 Die Beiträge im Überblick

Zu den Beiträgen der UHH gehört ein erheblicher technischer Aufwand zur Implementierung des DELIA-Software-Stacks, der Monitoring-, Wartungs- und Debugging-Tools sowie umfangreiche wissenschaftliche Forschung zu neuartigen Techniken zur Erhöhung der Ausfallsicherheit dienstleistungsorientierter und zeitkritischer Systeme. Unsere Beiträge lassen sich in den folgenden fünf Hauptkategorien zusammenfassen:

B1. Entwicklung des DELIA-Software-Stacks: Die UHH war hauptverantwortlich für die Entwicklung des DELIA-Software-Stack. Wir haben zunächst eine detaillierte Analyse der vorhandenen Hypervisoren und Containerisierungstechnologien für die Plattformvirtualisierung durchgeführt. Des Weiteren haben wir mehrere Betriebssysteme auf ihre Eignung für die DELIA-Plattform analysiert (AP1.3.UHH.1). In Abschnitt 3.1 werden die entsprechenden Ergebnisse kurz vorgestellt. Anschließend haben wir die erforderlichen Softwaremodule implementiert, um die Verwendung von virtuellen Xen-Maschinen zur Bereitstellung isolierter kritischer und nicht-kritischer

Domänen und von Linux-Containern (LXC) zum Deployment virtueller Dienste in beliebigen Domänen zu ermöglichen (AP 2.1.UHH.1, APP 2.2.UHH.1). Für alle genannten Softwarekomponenten wurden außerdem umfangreiche Protokollierungsmöglichkeiten, Debugging- und Testwerkzeuge entwickelt (AP 2.3.UHH.3). Neben den eigenständigen Softwaremodulen umfasste dieser Implementierungsaufwand auch die Schaffung notwendiger Schnittstellen, um sie insgesamt zu integrieren und ihre Code-Abhängigkeiten zu berücksichtigen. Wir haben mehr als 10.000 Zeilen Code (LoC) in verschiedenen Programmiersprachen geschrieben und Installationsanleitungen sowie weitere Dokumentation für diesen Quellcode erstellt. Abschnitt 3.2 beschreibt die wichtigsten Module, Schnittstellen und Werkzeuge, die wir implementiert haben, im Detail.

B2. Orchestrierung einer dienstorientierten Architektur und zeitabhängige Kommunikation: Der Entwurf eines serviceorientierten Systems mit einer Vielzahl von gemischt-kritischen Diensten erfordert die Platzierung, Wartung und Verbindung virtueller Dienstinstanzen innerhalb begrenzter Systemressourcen. Dementsprechend haben wir den DELIA-Supervisor implementiert, der für die Reservierung der benötigten Ressourcen auf den DELIA-Knoten verantwortlich ist und virtuelle Maschinen und Container für die gegebenen Serviceanforderungen initiiert. Er koordiniert nicht nur die Hauptfunktionen des Systems, sondern ist auch für die Neuverteilung der redundanten Ressourcen im Störungsfall zuständig. Zu diesem Zweck überwacht der Supervisor alle virtuellen Instanzen und repariert sie automatisch nach der Erkennung eines Ausfalls oder Angriffs, indem er die potenziell ausgefallenen oder gefährdeten Dienste migriert (AP 2.2.UHH.2, AP 2.3.UHH.1, AP 2.3.UHH.4). In Abschnitt 3.2 werden die Funktionen des Supervisors ausführlich beschrieben. Darüber hinaus haben wir mehrere Optimierungsmodelle für das optimale Deployment von Diensten und die Kommunikation zwischen den Diensten vorgeschlagen, die vom Supervisor zur Konfiguration der gesamten DELIA-Plattform verwendet werden können (AP 2.2.UHH.1). Diese Modelle werden in Abschnitt 3.3 vorgestellt. Ein zentraler Supervisor stellt jedoch Single-Point-of-Failure dar und kann die Skalierbarkeit und Autonomie des Gesamtsystems beschränken. Dahere haben wir auch dezentrale und verteilte Methoden zur (Selbst-)Konfiguration von Diensten und Kommunikation entwickelt, deren Einzelheiten in den Abschnitten 3.4 und 3.5 (AP 2.2.UHH.1, AP 2.3.UHH.4) beschrieben werden. Letzteres beinhaltet auch ein Optimierungsmodell zur zuverlässigen Konfiguration von IEEE 802.1Qbv Time-aware Shaper (TAS), dem Scheduling-Protokoll für zeitkritische Kommunikation, das von den Projektpartnern für die Interkommunikation von DELIA-Knoten implementiert wurde.

B3. Maßnahmen zur Erhöhung der Resilienz zeitkritischer Kommunikation: Avioniksysteme verwenden in der Regel redundante Komponenten und Verbindungen, um potenzielle Hardware-, Software- und Verbindungsausfälle nahtlos zu bewältigen. Der DELIA-Supervisor ist für die Verwaltung redundanter Ressourcen verantwortlich. Er erfordert jedoch auch die Konfiguration und Wartung der entsprechenden zeitkritischen Netzwerkprotokolle, nämlich IEEE 802.1CB Frame Replication and Elimination for Reliability (FRER) und IEEE 802.1aq Shortest Path Bridging. FRER ist das wichtigste Redundanzprotokoll für die Kommunikation der DELIA-Knoten untereinander. Dementsprechend haben wir eine Metrik für die Konfiguration von redundanten Pfaden gegen Link-Ausfälle in FRER (AP 2.2.UHH.1) vorgeschlagen. Außerdem haben wir Verbesserungen an den bestehenden Mechanismen dieses Protokolls vorgeschlagen, um seine zuverlässige Nutzung zu gewährleisten. Abschnitt 3.6 stellt die Details dieser Vorschläge vor. Diese Verbesserungen wurden auch in einem Simulations-Framework implementiert und als

Open-Source-Beitrag veröffentlicht.

B4. Sicherheitsmaßnahmen, Überwachung und Erkennung von Cyber-Angriffen: Während Resilienz- und Sicherheitsmaßnahmen die erste Verteidigungslinie gegen Angriffe darstellen, sind weitere Werkzeuge für die kontinuierliche Überwachung der Plattform, die Erkennung von Angriffen und Mechanismen zur Meldung von Vorfällen zur Erhöhung Resilienz des Gesamtsystems erforderlich. Dementsprechend haben wir zunächst die potenziellen Bedrohungen definiert, indem wir mehr als 30 Angriffe auf IEEE 802.1 TSN-Protokolle aufgelistet und potenzielle Angreifermodelle identifiziert haben (AP 1.2.UHH.1, AP 1.2.UHH.2). Abschnitt 3.7 stellt einige dieser Angriffe vor. Wir haben auch die vorhandenen Werkzeuge analysiert, um die Vertraulichkeit und Integrität der zeitkritischen Kommunikation zu gewährleisten und virtuelle Dienstinstanzen gegen böswillige Versuche zu härten. Unter Berücksichtigung potentieller Sicherheitsprobleme in MCS haben wir ein System zur Erkennung von Eindringlingen entwickelt, indem wir mit Zeek ein bestehendes Open-Source-Monitorin-Tool (AP 2.3.UHH.2) erweitert haben. Seine Implementierungsdetails und Funktionen werden in Abschnitt 3.8 kurz beschrieben. Auch dieser Beitrag wurde also Open-Source veröffentlicht.

Des Weitere nhaben wir ein Optimierungsmodell entwickelt, um Moving Target Defense (MTD)-Strategien gegen zielgerichtete Angriffe auf serviceorientierte Architekturen (AP 2.3.UHH.4) zu finden. Diese Strategien erschweren die Erkundung der Dienstkonfiguration auf den DELIA-Knoten und tragen dazu bei, die Kontrolle über kritische Dienste wiederzuerlangen, die potenziell gefährdet sind, noch bevor ein bösartiger Vorfall entdeckt wird. In Abschnitt 3.9 wird diese Strategie anhand eines Beispielszenarios beschrieben.

B5. Integration und Einsatz aller Komponenten: Schließlich haben wir eine Testumgebung aufgebaut, um alle unsere oben genannten Beiträge zu integrieren und alle implementierten Funktionen in einem Beispielszenario zu demonstrieren (AP 3.1.UHH.1, AP 3.2.UHH.1). Wir haben zudem auch Angriffs- und Ausfallszenarien implementiert, um das Gesamtsystem gegen die geplanten Bedrohungen zu testen (AP 2.3.UHH.3, AP.3.1.UHH.1). Es umfasst die Installation und Konfiguration der notwendigen TSN-Komponenten und virtualisierten Server. Die Einzelheiten des Testbeds und des Demonstrationsszenarios sind in Abschnitt 3.10 aufgeführt.

Tabelle 1: Die Gesamtbeiträge der UHH.

Contribution	**Action Point**												**Abschnitts**
	AP 1.2		**AP 1.3**	**AP 2.1**	**AP 2.2**		**AP 2.3**				**AP 3.1**	**AP 3.2**	
	1.2.1	**1.2.2**	**1.3.1**	**2.1.1**	**2.2.1**	**2.2.2**	**2.3.1**	**2.3.2**	**2.3.3**	**2.3.4**	**3.1.1**	**3.2.1**	
B1			✓	✓	✓				✓				Abschnitt 3.1, 3.2
B2					✓	✓	✓			✓			Abschnitt 3.2, 3.3, 3.4, 3.5
B3				✓									Abschnitt 3.6
B4	✓	✓						✓		✓			Abschnitt 3.7, 3.8
B5									✓		✓	✓	Abschnitt 3.10
	•	•	•	•	•	•	•	•	•	•	•	•	

Die Tabelle 1 zeigt die Zuordnung der oben genannten technisch-wissenschaftlichen Beiträge zu den jeweiligen Arbeitspaketen.

1.4 Projektbedingungen

Das Projekt zielte darauf ab, die erforderlichen Hardware- und Software-Module für den Entwurf einer widerstandsfähigen Verarbeitungs- und Kommunikationsplattform zu entwickeln. Der Aufbau einer offenen, flexiblen, störungs- und angriffsresistenten Plattform stellt das eher statische Design aktueller missionskritischer Systeme in Frage und erfordert einen erheblichen Entwicklungs- und Forschungsaufwand. Im Rahmen des Projekts war die UHH allein für die Entwicklung resilienter Systemfunktionen und neuartiger Resilienzmethoden verantwortlich und investierte auch erheblichen Aufwand für die Implementierung aller konzipierten Komponenten. Dies wurde mittels der Personalmittel und der über das Projekt geförderte TVL-E13-Stelle realisiert. Für die Publikation der 12 wissenschaftlichen Beiträge der UHH im Projekt sowie für Projekt-Meetings, wurden Teile der Reisekosten verwendet. Zum Aufbau eines lokalen Demonstrators wurde Hardware in Form von drei programmierbaren Switches, zwei programmierbaren Netzwerkkarten und zwei Tower-Server angeschafft. Diese Hardware wird nun über das Projekt hinaus weiter genutzt und in Forschung und Lehre eingesetzt.

1.5 Zusammenarbeit mit Konsortialpartnern

Im Rahmen des Projekts hat die UHH mit allen an DELIA beteiligten Projektpartnern zusammengearbeitet. UHH, AED, solectrix und ZAL haben die Anforderungen und die Implementierung von TSN-Protokollen und -Funktionalitäten ausgiebig diskutiert. Gemeinsam mit der Universität Stuttgart haben wir die vorhandenen Virtualisierungstechnologien und in Fragen kommenden Betriebssysteme analysiert und dazu auch einen technischen Report geschrieben. Außerdem haben wir zusammen die initiale Architektur der DELIA-Knoten festgelegt und gemeinsam über die erforderlichen Module und die Konfiguration der virtuellen Instanzen entschieden. UHH und ZAL haben zu einem gemeinsamen Code-Repository für die Entwicklung von DELIA-Modulen beigetragen. Außerdem haben wir bei der Auswahl von Demonstrationsanwendungen für das endgültige Testbed zusammengearbeitet.

2 Stand der Technik

Die UHH hat Entwurfs- und Orchestrierungsmethoden für eine dienstorientierte Architektur implementiert (in Bezug auf die Verarbeitungsfunktionen von DELIA) und Vorschläge für Zuverlässigkeits- und Sicherheitsmaßnahmen für zeitabhängige Kommunikation gemacht (in Bezug auf die Kommunikationsfunktionen von DELIA). Bei der Implementierung des DELIA-Software-Stacks haben wir die Technologien sorgfältig geprüft und sie entsprechend in das Projekt integriert (siehe Abschnitt 3.1). Die UHH hat während des gesamten Projektverlaufs neue wissenschaftliche Ansätze und Lösungen bewertet und kontinuierlich in das Projekt integriert. Im Projektverlauf gab es keine nennenswerten Weiterentwicklungen des Stands der Technik die hier explizit erwähnt werden müssen. Auf der Basis bestehender Arbeiten haben wir selber neuartige Lösungen entwickelt, die Forschungslücken in der Literatur schließen und damit die Hauptziele von DELIA ermöglichen. Nachfolgend wird der aktuelle Stand der Technik und bestehende Probleme zusammengefasst. Außerdem stellen wir kurz Forschungsprojekte mit Bezug zu DELIA vor.

2.1 Verteilung von Diensten und dienstübergreifende Kommunikation

Ein effizientes Dienstverteilungsschema erfordert eine genaue Ressourcenorchestrierung in Bezug darauf, wo, wann und wie viele Dienstinstanzen bereitgestellt werden [18, 19]. Außerdem wirken sich die Abhängigkeiten zwischen Diensten [20], die Migration von Diensten [21], der Lastausgleich zwischen Diensten [22] und das Task-Scheduling [23] auf die Kosten der Anbieter und die User-Experience aus. Die richtige Zuteilung dieser Dienste [24, 25] ist wichtig, um zum Beispiel die Betriebskosten [26] und

die Fragmentierung der physischen Ressourcen [27] für die Anbieter zu minimieren und die Dienstqualität [26] und die Reaktionsfähigkeit [28] für die Nutzer zu maximieren. Verschiedene andere Studien betrachten die Probleme der optimalen Dienstzuweisung und des optimalen Routings gemeinsam, um die Dienste auf Hosts und Pfaden [29, 30] zu verteilen und die Netzwerkressourcen optimal zu nutzen. Im Gegensatz zu bestehenden Arbeiten konzentrieren sich unsere im Rahmen von DELIA entwickelten Dienstbereitstellungssysteme auf neuartige virtualisierte und eingebettete Netzwerke. Da die Kommunikation zwischen den Diensten definiert ist, sind die Beziehungen zwischen den Diensten entscheidend für unser Netzwerkdesign, das sowohl das Deployment von Diensten als auch das Traffic Engineering berücksichtigt. Daher handelt es sich um ein gemeinsames Problem der Dienstzuweisung und des dienstübergreifenden Routings, wobei das Routing auch von der Dienstzuweisung abhängt. Darüber hinaus wird das Problem noch schwieriger, wenn man als zusätzliche Anforderung Ausfallsicherheit zu einem solchen dynamischen Deployment-Schema hinzufügt.

2.2 Zuverlässigkeit und Qualität der Dienste im TSN

Von den verschiedenen IEEE 802.1 TSN-Standards sind die Redundanz- und Zeitplanungsprotokolle FRER und TAS für DELIA besonders relevant.

Bei der dynamischen Planung zeitempfindlicher Datenströme gehen die meisten Arbeiten davon aus, dass die Routingpfade im Voraus bekannt sind, und lassen die Pfad(neu)konfiguration außer acht. In [31] schlagen die Autoren einen Mechanismus zur Ressourcenzuweisung vor, um Datenströme zur Laufzeit unterzubringen und zu migrieren. Der potenzielle Migrations-Overhead (z. B. die Ende-zu-Ende-Latenz und die Anzahl der rekonfigurierten Pfade) wird jedoch nicht analysiert. Nur wenige Veröffentlichungen behandeln das Scheduling zusammen mit dem Pfadberechnungsproblem [32, 33]. Unsere Arbeit überschneidet sich teilweise mit diesen Studien, indem sie Routing-Pfade mit Zeitplänen kombiniert. Im Gegensatz zu den bestehenden Modellen, die Pfadzuweisungsprobleme mit eher einfachen Metriken wie Pfadlängen und Link-Gewicht lösen, umfasst unser Modell zusätzlich auch die Gate-Konfiguration als TSN-spezifischen Aspekt. Für Fehlertoleranz und Zuverlässigkeit von zeitkritischen Streams durch Redundanz vergleichen die Autoren in [34] die Vor- und Nachteile von zwei verschiedenen Protokoll-Stack-Designs für Redundanzprotokolle: entkoppelt mit oder integriert in die Stream-Reservierung. [35] untersucht den domänenspezifischen Einsatz von FRER mit Fokus auf Echtzeitkomponenten in der Industrie 4.0. In [36] wird ein detaillierter Überblick über FRER gegeben. Außerdem implementieren die Autoren ein standardkonformes Modell und verifizieren die Wirksamkeit von FRER, ohne dabei auf numerische Ergebnisse einzugehen.

Die Autoren von [37] stellen den Einsatz von FRER in verschiedenen Topologien vor und diskutieren praktische Fragen und Funktionsprinzipien. In [38] erörtern die Autoren die Grenzen von FRER und gehen insbesondere auf die Pufferdimensionierung für Sequenznummern, die Unzulänglichkeit des Fehlerrückkopplungsmechanismus, die Zustellung außerhalb der Reihenfolge, die zusätzliche Netzwerklast und die empfindliche Konfiguration ein.

Obwohl seine Dynamik, bestimmte Einschränkungen und die Koexistenz mit anderen zeitbasierten TSN-Protokollen in einigen Studien diskutiert werden, wurden die Auswirkungen der Pfadauswahl und die Verwendung eines Eliminierungsmechanismus für FRER noch nicht untersucht. Dementsprechend haben wir uns auf diese konzentriert.

2.3 Monitoring von TSN auf Cyber-Angriffe

Es gibt bereits mehrere Ansätze zum Schutz zeitabhängiger Kommunikation vor Cyber-Angriffen die alle auf bestehenden TSN-Protokollen aufbauen. In [39] verwenden die Autoren zum Beispiel

IEEE 802.1Qav Credit-based Shaper (CBS), um Denial-of-Service-Angriffe zu verhindern. Die Autoren von [40] und [41] kombinieren das IEEE 802.1Qci Per-Stream Filtering and Policing (PSFP)-Protokoll [42] mit einem zentralisierten Controller, um Ingress-Policies für Paket-Ankunftszeiten und -Raten sowie Stream-Bandbreite durchzusetzen. Ähnlich wird in [43] die Effektivität von PSFP zum Schutz zeitkritischer Automobilnetzwerke analysiert. In [44] schließlich diskutieren die Autoren die Umseztung von Sicherheits-Policies mittels PSFP, die von einem zentralisierten Policy-Server durchgesetzt werden.
Es gibt auch praktischere Design- und Implementierungsbemühungen für die Überwachung und den Schutz zeitempfindlicher Systeme. In [45] schlagen die Autoren ein Monitoring-System für den Status von TSN-Bridges und Links sowie die Genauigkeit der Zeitsynchronisation vor, ohne dabei aber explizit Sicherheitsaspekte zu berücksichtigen. Muguira et al. [46] schlagen ein Sicherheitsmodul für Hardware-Verschlüsselung und Authentisierung zur Verbesserung von TSN-Protokollen vor. Der IEEE-Standard 802.1AE Media Access Control (MAC) ermöglicht Integrität und Vertraulichkeit für Ethernet-Datenverkehr umzusetzen und kann mit anderen TSN-Standards kombiniert werden [47].
Keine der oben genannten Arbeiten bietet ein dediziertes Monitoring von IEEE TSN-Protokolle auf Fehler und Cyber-Angriffe. Im Gegensatz dazu haben wir im Teilprojekt mit *TSN-Zeek* eine quelloffene und erweiterbare IDS-Lösung umgesetzt, die TSN-spezifische Angriffe im Kontext von DELIA erkennen kann.

2.4 Projekte mit Bezug zu DELIA

Es gibt laufende Standardisierungsaktivitäten wie die Automotive Virtual Platform Specification [48] und Future Airborne Capability Environment (FACE) [49], die die Nutzung von Open-Source-Virtualisierungstechnologien in kritischen Fahrzeug- und Militärsystemen vorbereiten. Ein vom BMWi gefördertes Projekt, *IMMUNE*, zielte darauf ab, SDN-basierte Rekonfigurations-, Überwachungs- und Sicherheitslösungen insbesondere für industrielle Systeme der nächsten Generation zu entwickeln [50]. *secVI* ist ein weiteres vom BMBF gefördertes Projekt, das sichere In-Vehicle-Netze unter Verwendung von SDN- und TSN-Technologien entwickelt [51]. Ein weiteres aktuelles Projekt, *AIDA*, das von der schwedischen Knowledge Foundation finanziert wird, konzentriert sich auf die Rekonfigurierbarkeit von TSN sowie auf die Optimierung von containerbasierten Diensten [52].
Obwohl diese Projekte die Forschungsprobleme in ähnlichen kritischen Domänen unter Verwendung ähnlicher Technologien, d.h. SDN, TSN und Virtualisierung, angehen, zielt keines von ihnen auf eine holistische Lösung inklusive Maßnahmen zur Erhöhung der Resilienz ab. DELIA verfolgt mit der Kombination aus dem Einsatz dynamischer virtueller Dienste und neuartigen TSN-Technologien einen solchen holistischen Ansatz und zeigt wie diese Technologien für erhöhte Fehlertoleranz und Resilienz gegenüber Cyber-Angriffen nutzbringend eingesetzt werden können.

3 Technische und wissenschaftliche Beiträge

Dieser Abschnitt beschreibt die in Abschnitt 1.3 kurz skizzierten Hauptbeiträge der UHH im Detail.

3.1 Analyse der vorhandenen Virtualisierungs- und Netzwerktechnologien

Um die am besten geeigneten Virtualisierungstechnologien für den Aufbau einer service-orientierten Kommunikations- und Prozessorplattform zu finden, haben wir Hypervisoren und Open-Source-Betriebssysteme (OS) auf ihre Eignung geprüft und eine erste Architektur für den DELIA-Knoten entwickelt. Zunächst haben wir allgemeine Anforderungen an den DELIA-Knoten definiert, wie z. B. die Unterstützung mehrerer Betriebssysteme, die Verwendung von Open-Source-Software, die Anpass-

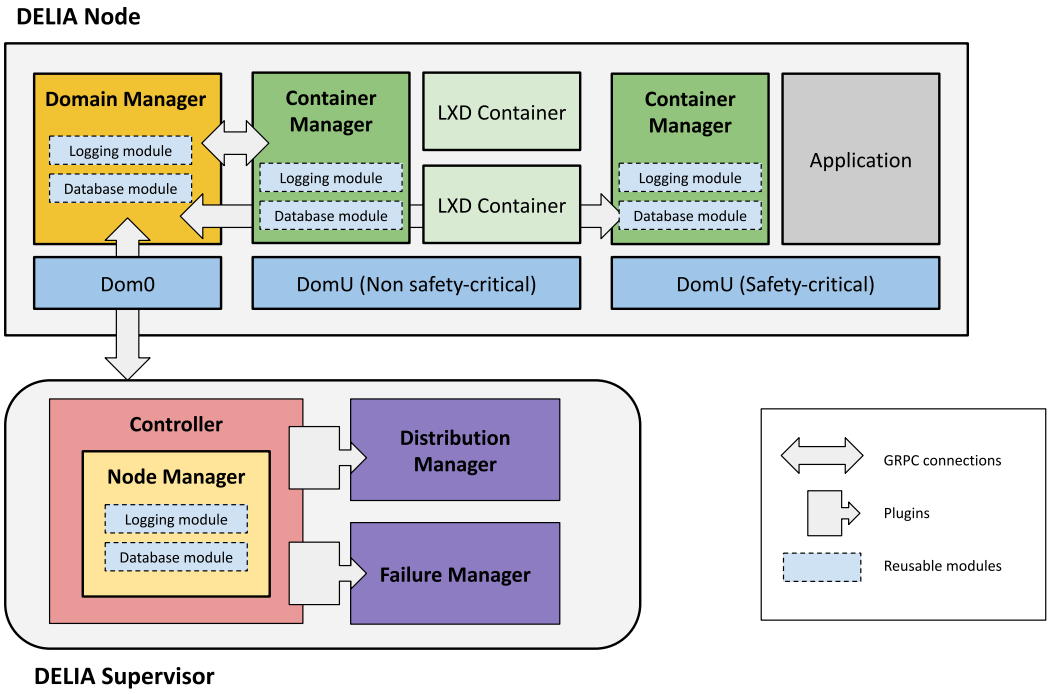

Abbildung 1: Die Hauptkomponenten des DELIA-Software-Stacks.

barkeit und Konfigurierbarkeit sowie Zertifizierbarkeit. Anschließend führten wir einen detaillierten Vergleich bestehender Hypervisoren und Betriebssysteme in Bezug auf ihre Typen, Open-Source-Verfügbarkeit, Standardkonformität und Plattformunterstützung durch. Schließlich wurde Xen als Hypervisor-Technologie für die DELIA-Knoten ausgewählt, um darüber sowohl kritische als auch unkritische virtuelle Maschinen zu betreiben.

Als Betriebssystem wurde Alpine für die DELIA-Knoten ausgewählt und um die notwendigen Tools für den implementierten Software-Stack erweitert. Diese Arbeiten haben wir in einem Technischen Report zusammengefasst [1].

Neben der Auswahl der Basistechnologien und einer Architektur für die DELIA-Knoten haben wir auch potenzielle Netzwerkprotokolle für die Netzwerkerkennung und -verwaltung zwischen miteinander verbundenen Komponenten geprüft. Wir sind zu dem Schluss gekommen, dass das bestehende Ethernet-Verwaltungsprotokoll IEEE 802.1aq Shortest Path Bridging (SPB) die am besten geeignete Lösung ist, da sie sich auch direkt mit den IEEE 802.1 TSN-Protokollen kombinieren lässt. Wir haben im Kontex des Projekts einen ausführlicheren technischen Report über SPB [2] verfasst.

Schließlich haben wir ein Positionspapier über die proaktiven und reaktiven Resilienzmechanismen von virtualisierten eingebetteten Netzen veröffentlicht [3].

3.2 Implementierung des DELIA-Software-Stacks

Die UHH war hauptverantwortlich für die Implementierung des DELIA-Software-Stacks, der die Arbeitsroutinen, die notwendigen Schnittstellen und die Konfiguration der virtuellen Dienste auf den DELIA-Knoten definiert. Außerdem hat die UHH den DELIA-Supervisor implementiert, der virtuelle Dienste initiiert und orchestriert. Abbildung 1 zeigt die Hauptkomponenten des Software-Stacks auf einem DELIA-Knoten und einen Supervisor.

Für die Umsetzung des DELIA-Software-Stack hat die UHH hier mehr als 10.000 Zeilen Quellcode geschrieben, darunter einige Python-Skripte zur Implementierung der Gesamtmodule (bunte Kästchen),

Schnittstellen (graue Pfeile) und Bash-Skripte zur Konfiguration der unterschiedlichen Module. Des Weiteren wurden Test- und Debugging-Tools umgesetzt. Die Code-Basis und die Installationsanleitung sind mittels git[1] verfügbar.

Implementierung des DELIA-Knotens Die Software eines DELIA-Knoten läuft in einem vorkonfigurierten Alpine-Betriebssystem, das auf dem Xen-Hypervisor aufsetzt. In Abbildung 1 stellt dom0 die Kontrolldomäne des Knotens dar und beherbergt den von uns implementierten **Domain-Manager**. domUs sind kritische und sicherheitskritische Gast-*Domains*, d. h. virtuelle Xen-Instanzen, die eine andere Version des vorkonfigurierten Alpine-Betriebssystems verwenden, das vom Domain-Manager orchestriert wird. Der Domain-Manager implementiert eine Schnittstelle, die die vorhandenen Xen-Verwaltungstools erbt. Er ist hauptsächlich für das Erstellen, Entfernen, Umschalten und die Statusprüfung von Xen-Instanzen sowie für die Protokollierung aller domänenbezogenen Ereignisse zuständig.
Jede Domäne beherbergt auch eine Reihe von Diensten in Form von Linux-Container (LXC/LXD), die von ihrem jeweiligen **Container-Manager** verwaltet werden. Ähnlich wie der Domain-Manager implementiert der Container-Manager eine Schnittstelle für LXD-Verwaltungstools, um dienstbezogene Funktionen auszuführen. Er ist verantwortlich für das Erstellen, Entfernen und Migrieren von LXD-Containern sowie für die Protokollierung aller containerbezogenen Ereignisse. Beide Manager nutzen die gleichen implementierten Protokollierungsmöglichkeiten und Datenbankmodule.
Alle Funktionen der Domain- und Container-Manager werden vom DELIA-Supervisor ausgelöst. Dazu verwendet der Supervisor den **Node-Manager**, dessen Einzelheiten im folgenden Abschnitt beschrieben werden. Der Supervisor und die Domänen- und Container-Manager kommunizieren über dedizierte GRPC-Kanäle. Jeder dieser Manager betreibt einen GRPC-Server und fungiert auch als GRPC-Client, um eine unidirektionale Kommunikationskette zu bilden. In dieser Kette kann der Controller sowohl den Domänen- als auch den Container-Manager befehligen, und der Domänen-Manager kann den Container-Manager befehligen. Es ist zu beachten, dass der Supervisor sich zuerst mit dem jeweiligen Domänen-Manager verbinden muss, um einen beliebigen Container-Manager in einer der Domänen des jeweiligen DELIA-Knotens zu erreichen. Diese unidirektionale und indirekte Verbindung schränkt den Zugriff der einzelnen Dienste auf den Supervisor ein, hält sie aber dennoch als eigenständige Instanzen, die nur über RPC-Aufrufe verbunden werden können.

Implementierung des DELIA-Supervisors Der DELIA-Supervisor besteht aus einem Controller, einem Distribution-Manager und einem Failure-Manager. Der Controller erweitert den Node-Manager, der die grundlegenden Funktionen zur Orchestrierung von Domänen und Containern auf DELIA-Knoten bereitstellt, indem er mit den jeweiligen Domänen- und Container-Managern kommuniziert. Darüber hinaus ist er für regelmäßige Heartbeats von DELIA-Knoten, das Abrufen von Protokollen und das Auslösen notwendiger Mitigations- oder Failover-Funktionen im Falle eines Angriffs oder eines Ausfalls eines Teils des Gesamtsystems verantwortlich. Um diese Funktionen bereitzustellen, implementiert der **Distribution-Manager** die Ressourcenreservierungslogik für Domänen und Container. Er löst entweder ein Optimierungsproblem oder führt Greedy-Heuristiken aus, um über die Dienstkonfiguration zu entscheiden, d. h. darüber, welche DELIA-Knoten welche Dienste unter Berücksichtigung der Dienstanforderungen und der verfügbaren Knotenressourcen hosten sollen. Es wird hauptsächlich für das anfängliche Deployment von Diensten auf der DELIA-Plattform verwendet. Das Optimierungsmodell kann auch redundante Ressourcen reservieren, die im Falle eines Fehlers verwen-

[1] `https://git.informatik.uni-hamburg.de/iss/delia/src`

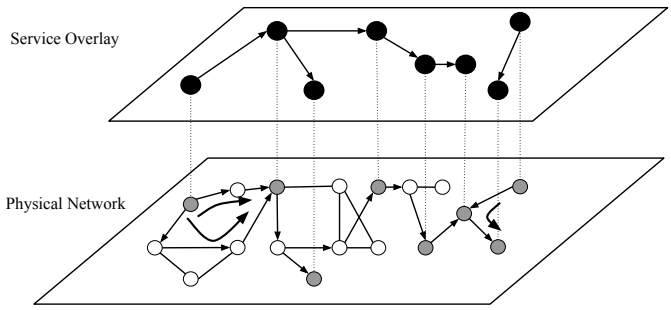

Abbildung 2: Ein Beispiel für einen Dienst-Overlay.

det werden. Die Grundlagen des Optimierungsmodells sind in Abschnitt 3.3 dargestellt. Der **Failure-Manager** entscheidet über die zu ergreifenden Maßnahmen für den Fall, dass der Controller einen Ausfall feststellt, z. B. aufgrund fehlender Heartbeats. Er kann die Migration von einzelnen Containern und Domänen zwischen Knoten oder deren Umverteilung auf mehrere Knoten unter Verwendung einer Reihe von Heuristiken festlegen.
Der Supervisor unterhält eine unidirektionale Verbindung mit allen Knoten, d. h. mit dem Domänenmanager jedes Knotens. Daher fungiert das Node-Manager-Modul nur als GRPC-Client und hostet keinen GRPC-Server wie die Domänen- und Container-Manager. Es kann völlig autonom arbeiten oder eine Konfigurationsdatei übernehmen, die ein bestimmtes Diensteinsatz- und Kommunikationsschema festlegt.

3.3 Entwicklung von Modellen für eine optimale und widerstandsfähige dienstorientierte Architektur

Die dienstorientierte Architektur der DELIA-Plattform erfordert die Einbettung eines Overlay-Netzes von Diensten in ein zugrunde liegendes physisches Netz von DELIA-Knoten, so dass die sich daraus ergebende Zuordnung den dienstübergreifenden Datenverkehr, die Latenz und vor allem die Anforderungen an die Ausfallsicherheit erfüllt.
Abbildung 2 zeigt ein Beispiel für die Einbettung eines Dienst-Overlays (schwarze Knoten) in das zugrunde liegende physische Netz (graue Knoten). Eine Verbindung zwischen zwei Diensten (eine Kante zwischen zwei schwarzen Knoten) stellt eine Kommunikationsanforderung mit bestimmten Bandbreiten- und Latenzanforderungen dar. Die Verbindung zwischen zwei physischen Knoten (eine Kante zwischen zwei grauen Knoten) ist eine physische Verbindung mit einer nominalen Bandbreitenkapazität. Eine Dienstinstanz kann einem einzelnen Knoten zugewiesen werden, um die Kommunikation mit anderen Knoten herzustellen, die benachbarte Dienstinstanzen beherbergen. Das Gesamt-Deployment sollte durch (i) die von den Diensten beanspruchten Knotenressourcen und (ii) die für die Anforderungen zwischen den Diensten erforderlichen Verbindungskapazitäten begrenzt werden. Darüber hinaus stellt die Verzögerung auf dem Pfad zwischen zwei kommunizierenden Diensten ein weiteres Constraint dar.
Dementsprechend haben wir das JSAR-Modell (Joint Service Allocation and Routing) vorgeschlagen, um das obige Problem der Diensteinbettung als gemischt-ganzzahliges lineares Programmiermodell (MILP) zu formulieren. Es liefert schließlich das optimale Deployment von Diensten und das optimale Routing von Datenflüssen zwischen diesen im Hinblick auf die Ressourcennutzung und die Dienstqualität. Dieses Modell wurde zudem um weitere Constraints für die Erhöhung der Fehlertoleranz erweitert. Schließlich haben wir mehrere Heuristiken entwickelt, um ein Failover auf der Basis von

vorreservierten redundanten Ressourcen umzusetzen. JSAR, die Fehlertoleranz-Erweiterungen und die Heuristiken wurden zudem direkt für den Verteilungsmanager des DELIA-Supervisors implementiert. Die Modelle und ihre Evaluation wurden auf einer hochrangingen Tagung [4] publiziert.
Das JSAR-Modell selber ist jedoch zu komplex, um darüber größere Probleme mit größeren Netzen und größeren Dienste-Overlays zu lösen. Außerdem kann es schwierig sein, innerhalb knapper, vorgegebener Systemressourcen genügend redundante Ressourcen für den Fall eines einzelnen Knotenausfalls zu finden. Daher haben wir die Erweiterung JSAR-SP vorgeschlagen. Hier teilen sich mehrere Kommunikations-Demands die gleichen redundanten Backup-Ressourcen, insofern diese nicht vom gleichen Ausfallszenario betroffen sind. JSAR-SP verbessert auch die Formulierung des Optimierungsmodells von JSAR und ermöglicht so eine deutlich schnellere Lösungszeit. Wir haben das JSAR-SP-Modell, seine Analyse und Bewertung in einem hochrangingen Journal [5] veröffentlicht.

3.4 Verteilte Konfiguration einer service-orientierten Architektur

Der DELIA-Supervisor führt alle Funktionen der Dienstverteilung und Netzkonfiguration zentral durch. Darüber hinaus war die UHH auch dafür verantwortlich, potenzielle dezentrale und verteilte Methoden zu erforschen, die das System zur Selbstkonfiguration und Autonomie befähigen. Dementsprechend haben wir einen bio-inspirierten Ansatz vorgeschlagen, der ein Konsensschema zwischen DELIA-Knoten unter Verwendung probabilistischer Ant-Colony-Optimization (ACO)-Funktionen [53] definiert, so dass das Gesamtdesign und die Wartung der dienstorientierten Architektur ohne zentralen Controllers auskommt.
Der bio-inspirierte Ansatz umfasst drei aufeinanderfolgende Phasen: Dienstzuweisung, Routing und Umverteilung. Bei der Formulierung der einzelnen Phasen haben wir die Safety-Anforderungen eines missionskritischen Systems wie DELIA berücksichtigt, indem wir (i) das Deployment von mehreren kritischer Dienste auf denselben physischen Knoten vermeiden und (ii) die Verkehrslast ausgleichen, um die Demand-Zuweisung zu erweitern, anstatt sich auf die dedizierten kürzesten Wege zu verlassen. Beides verhindert Single-Point-of-Failure und Performance-Bottlenecks, die zu Unterbrechungen kritischer Dienste und zu Unterbrechungen des Datenverkehrs führen können.
Bei der Evaluierung unseres Frameworks haben wir festgestellt, dass es möglich ist, mit unserem verteilten Framework viel schneller ein nahezu optimales Ergebnis in Bezug auf die Knoten- oder Verbindungsressourcennutzung zu erzielen als mit JSAR-Modellen, die wir für einen zentralen DELIA-Supervisor vorgeschlagen haben. Allerdings waren die zentralisierten JSAR-Modelle erfolgreicher bei der Mehrzieloptimierung, d. h. bei der gleichzeitigen Optimierung der Knoten- und Verbindungsauslastung. Wir fanden auch heraus, dass verschiedene Umverteilungsheuristiken verwendet werden können, die den Kompromiss zwischen Ressourcenauslastung und Verteilungsdauer berücksichtigen. Unsere Ergebnisse haben wir in einer weiteren wissenschaftlichen Veröffentlichung dokumentiert [6].

3.5 Autonome Konfiguration von zeitsensitiven Netzen

IEEE 802.1Qbv TAS ist eines der bekanntesten TSN-Protokolle für deterministische End-to-End-Kommunikation und wurde dazu auch in DELIA eingesetzt. Da die Konfiguration von TAS größtenteils von (i) manueller Ressourcenzuweisung und -planung und (ii) aktiver Beteiligung der TSN-Talker zur Deklaration ihrer Demands abhängt, ergibt sich eine erhebliche Abhängigkeit von einem Administrator/Controller und von End-Hosts die einbezogen werden müssen. Um damit umzugehen, haben wir das zeitabhängige optimale Routing (TSOR)-Optimierungsmodell auf der Basis eines Software-defined Networking (SDN)-Frameworks, SC-TSN, vorgeschlagen. Wir haben die Prinzipien für die Integration von SDN und TSN auf einer Tagung [7] vorgestellt.

SC-TSN beseitigt die End-Host-bezogenen Abhängigkeiten der TSN-Konfiguration. Anstatt explizite Ressourcenanfragen für zeitkritische Streams zu erwarten, lernen die modifizierten TSN-Brücken von SC-TSN automatisch die Verkehrscharakteristika. Sie unterstützen dann einen zentralen Controller, z. B. den DELIA-Supervisor, bei der Zuteilung ausreichender Ressourcen und der Planung zeitempfindlicher Streams über IEEE 802.1Qbv TAS in jeder TSN-Bridge in den Systemen. Hier nutzt der zentrale Controller unser `TSOR`-Modell, um die optimale Konfiguration der Gate-Control-Listen der TAS zu finden. `TSOR` ist ein gemischt-ganzzahliges lineares Programmiermodell, das die gesamte Ende-zu-Ende-Latenz der zeitabhängigen Kommunikation mit einer optimalen Gatterlistenkonfiguration minimiert. Unsere Ergebnisse zeigen, dass SC-TSN bei einer vernachlässigbaren Verzögerung eine optimale Ressourcenreservierung und -planung durchführen und erfolgreich mittlere bis große Mengen an zeitkritischen Streams in einer gegebenen physikalischen Topologie platzieren kann. Weitere Details finden Sie in der Originalarbeit [8].
Wir haben `TSOR` weiter genutzt, um dynamische Rekonfigurationsstrategien für zeitkritische Datenströme zu entwickeln, die eine optimale Dienstgüte und eine effiziente Ressourcennutzung gewährleisten. Diese Strategien reagieren auf sich ändernde Netzwerkbedingungen und -anforderungen, indem sie neue Streams zuweisen und bestehende Streams entfernen oder migrieren. Wir haben insgesamt vier Strategien mit unterschiedlichen Rekonfigurationshäufigkeiten, Auslösern und der Menge der rekonfigurierten Streams vorgeschlagen.
Die wichtigste Erkenntnis aus unserer Bewertung ist, dass verschiedene Rekonfigurationsstrategien je nach verfügbaren Netzwerkressourcen für mehr Effizienz oder je nach Kritikalität und strengen Anforderungen der Streams für eine bessere QoS ausgewählt werden können. Eine detailliertere Analyse dieser Strategien ist in der Originalarbeit [9] zu finden.

3.6 Zuverlässige Konfiguration von IEEE 802.1CB FRER

Die effektive Konfiguration von FRER ist entscheidend, um eine zuverlässige und fehlertolerante Kommunikation zu gewährleisten. In FRER können beliebige Pfade zwischen zwei Knoten für redundante Kommunikation genutzt werden. Dies steht im Gegensatz zu bestehenden traditionellen Protokollen, die bestimmte topologische Strukturen erzwingen. Schlecht ausgewählte Pfade können jedoch sowohl die Zuverlässigkeit als auch die Effizienz der Kommunikation beeinträchtigen. Wenn sich beispielsweise die konfigurierten Backup-Pfade auf einer gemeinsamen TSN-Bridge kreuzen, könnte die FRER-Eliminierungsfunktion alle doppelten Frames mit der gleichen Sequenznummer nach der Weiterleitung der ersten ankommenden Kopie verwerfen. Dies führt letztendlich zu einer Verschlechterung des gewünschten Redundanzgrades.
Um dies zu vermeiden, schlagen wir zunächst eine Pfadauswahlmetrik, `Reassurance`, sowie eine Verbesserung des Mechanismus zur Elimination von Frames vor, um solche unbeabsichtigten Eliminationen redundanter Frames zu verhindern und die Fehlertoleranz gegenüber zufälligen Verbindungsausfällen zu maximieren. `Reassurance` hilft dabei, die optimale Menge von k sich kreuzenden Pfaden auszuwählen, die die Anzahl der geschützten Verbindungen gegen bis zu $k-1$ Verbindungsausfälle maximiert. Unsere Verbesserung des Frame-Eliminierungsmechanismus macht TSN-Bridges auf potenzielle Kreuzungen redundanter Pfade aufmerksam, so dass sie zwischen tatsächlichen redundanten FRER-Frames und potenziell bösartigen oder fehlerhaften Paketen unterscheiden können. Während sie die redundanten Pakete bis zum konfigurierten Redundanzgrad weiterleiten, verwerfen sie weitere fehlerhafte Duplikate. Verschiedene Beispielszenarien, eine Analyse der `reassurance` und Implementierungsdetails unserer Verbesserungen im Frame-Replikationsmechanismus finden Sie im Originalbeitrag [10].

Obwohl FRER alle notwendigen Funktionen für eine redundante Kommunikation bereitstellt, schreibt er einer TSN-Bridge keine bestimmte Konfiguration dieser Funktionen vor. Das heißt, jede Funktion kann aktiviert oder deaktiviert werden, oder diese Funktionen können je nach Entwurf in unterschiedlicher Reihenfolge ablaufen. Nicht nur die interne Konfiguration, sondern auch die externe Konfiguration des FRER ist nicht Bestandteil der Norm. Daher ist für die Ermittlung alternativer redundanter Pfade und deren praktische Konfiguration für verschiedene zeitkritische Datenströme nach wie vor ein externer Mechanismus zur Ermittlung und Verwaltung des Netzes oder eine manuelle Konfiguration erforderlich. Dementsprechend untersuchen wir in dem Papier [11] die besten Verfahren für die interne Konfiguration von FRER und schlagen externe Protokolle für die Pfadermittlung und -konfiguration vor. Außerdem haben wir ein Open-Source-Simulations-Framework veröffentlicht, um das Testen verschiedener FRER-Konfigurationen in alternativen Implementierungen zu erleichtern. Letztendlich helfen diese Beiträge dabei, die fehlertolerante Kommunikation in DELIA optimal zu konfigurieren und ermöglichen es uns, den alternativen Einsatz in verschiedenen Szenarien von Verbindungsausfällen zu simulieren.

3.7 Erforschung von Angriffsvektoren gegen TSN-Protokolle

Die Sicherheitsbedrohungen für IEEE 802.1 TSN-Protokolle überschneiden sich mit mehreren Angriffsvektoren auf ältere Protokolle und Mechanismen in deterministischen und Echtzeitsystemen. Hier analysieren wir verschiedene TSN-Bedrohungen aus der akademischen Literatur und den Industriegesprächen und erweitern diese um potenzielle TSN-spezifische Schwachstellen. Diese Bedrohungen werden anhand des bekannten Bedrohungsmodellierungsrahmens STRIDE [54] in die wichtigsten TSN-Mechanismen eingeteilt.
Eine detaillierte Beschreibung all dieser Bedrohungen findet sich in der Originalarbeit [12]. Die Bedrohungen für SRP und FRER könnten besonders schwerwiegend sein und von jedem Angreifer ausgeführt werden, der einfach Pakete in das Netzwerk einschleusen kann. Daher haben wir uns in unserem Intrusion Detection System, das im nächsten Abschnitt vorgestellt wird, speziell mit diesen Bedrohungen beschäftigt.

3.8 Implementierung eines Systems zum Monitoring und zur Erkennung von Cyber-Angriffen

Während bestehende Sicherheitsmaßnahmen bereits gegen bestimmte Angriffsvektoren schützen können, erfordern TSN-spezifische Bedrohungen eine neuartige Lösung zur Überwachung und Erkennung von Cyber-Angriffen. Diese Lösung sollte in der Lage sein, (i) die neuen und erweiterten Frame-Strukturen der TSN-Protokolle wie IEEE 802.1 FRER und IEEE 802.1 SRP zu verarbeiten und (ii) übliches Verhalten und Muster dieser Protokolle logisch zu interpretieren. Dementsprechend haben wir mit *TSNZeek* ein System zur Überwachung und Erkennung von Cyber-Angriffen entwickelt, dass die neuen TSN-Protokolle analysieren und eine Reihe von TSN-spezifischen Angriffen erkennen kann. TSNZeek erweitert das Open-Source-Monitoring-Tool Zeek, das bereits für die Überwachung von Netzen und für Intrusion Detection eingesetzt wird.
TSNZeek besteht aus den Komponenten zum Monitoring und Erkennung von Angriffen, die in Abbildung 3 dargestellt sind. Die Monitoring-Komponente umfasst die Engines zur Verarbeitung und Protokollierung des empfangenen Netzwerkverkehrs. Die Intrusion-Detection-Komponente ist mit der Monitoring-Komponente verbunden und implementiert die Angriffserkennungslogik für TSN-spezifische Angriffe.
TSNZeek kann viele Bedrohungen erkennen, die wir in [12] identifiziert haben. Außerdem hat unsere

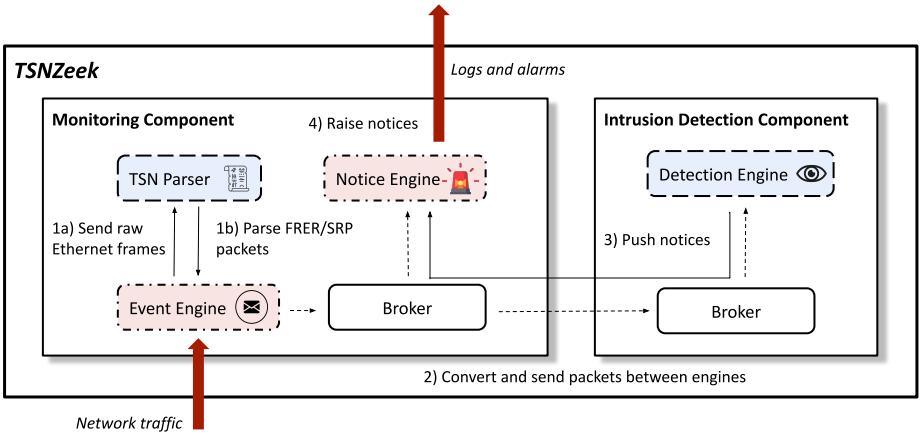

Abbildung 3: Die Architektur von TSNZeek

Evaluierung gezeigt, dass es einen vernachlässigbaren Overhead bei der Paketverarbeitung verursacht. Es kann zentral oder verteilt im DELIA-System und auch als virtueller Dienst platziert werden. Auch dieser Beitrag wurde auf einer Tagung veröffentlicht [13].

3.9 Moving target defense (MTD) mittels dynamischer Rekonfiguration von Diensten

Herkömmliche missionskritische Systemen können von Angreifern vor einem Angriff lange Zeit beobachtet werden, um z.B. Schwachstellen ausfindig zu machen. Der eigentliche Angriff kann dann innerhalb kürzester Zeit durchgeführt werden und lässt den Verteidigern nur ein kleines Zeitfenster zur Erkennung. Moving Target Defense (MTD) ermöglicht es den Verteidigern, kritische Bestandteile eines Systems, z. B. kritische Dienste in der DELIA-Plattform, neu zu konfigurieren, um die durch langfristige Beobachtung gewonnene Systemkenntnis des Angreifers obsolet zu machen und so potenzielle Angriffe zu erschweren. Abbildung 4 veranschaulicht eine MTD-Strategie, die von einer serviceorientierten Architektur profitiert. Auf den physischen Knoten (N1, N2, N3) werden verschiedene Arten von Hypervisoren (H1 und H2) und virtuellen Betriebssystemen (OS1 und OS2) eingesetzt. S1 und S8 (gehostet in N1 und N2) kommunizieren, um gemeinsam eine kritische Funktion auszuführen. In der Abbildung greift der Angreifer OS1 und H1 an, um S1 (Angriff 1a) zu deaktivieren. Alternativ kann er auch den co-residenten Dienst S2 angreifen, um instanzübergreifende Angriffe durchzuführen, oder passiv die Verbindung zwischen N1-N2 belauschen, um die Muster der kritischen Kommunikation zwischen S1-S8 (Angriff 1b) zu extrahieren.

Die Entdeckung von Schwachstellen und die entsprechende Entwicklung und Bereitstellung von Exploits für kritische Systeme erfordern viel Zeit und Ressourcen. Daher ist die Wahrscheinlichkeit eines erfolgreichen Angriffs umso höher, je länger das System in seiner statischen Konfiguration verbleibt. Im Beispiel-Angriffsszenario kann ein Verteidiger (i) S1 von OS1- und H1-spezifischen Schwachstellen abkoppeln und (ii) die Kommunikation zwischen S1-S8 sichern, die vom Angreifer physisch abgehört wird. Dementsprechend migriert der Angreifer S1 auf N2 (Verteidigung 2a), auf dem ein anderer Hypervisor und ein anderes Betriebssystem, H2 und OS2, laufen. Dadurch werden die bisherigen Bemühungen, S1 über H1 und OS1 anzugreifen, zunichte gemacht und die Zeit des Angreifers für die Entdeckung, Injektion und Ausnutzung eingeschränkt. Ergänzend wird die Kommunikation zwischen S1 und S8 so umkonfiguriert, dass sie nach der Migration von S1 (Verteidigung 2b) zwischen N2 und N3 hergestellt wird. Dadurch wird verhindert, dass der Angreifer eine konsistente Sicht auf die Kommunikation hat.

Dementsprechend haben wir ein integriertes Optimierungsmodell entwickelt, um optimale MTD-

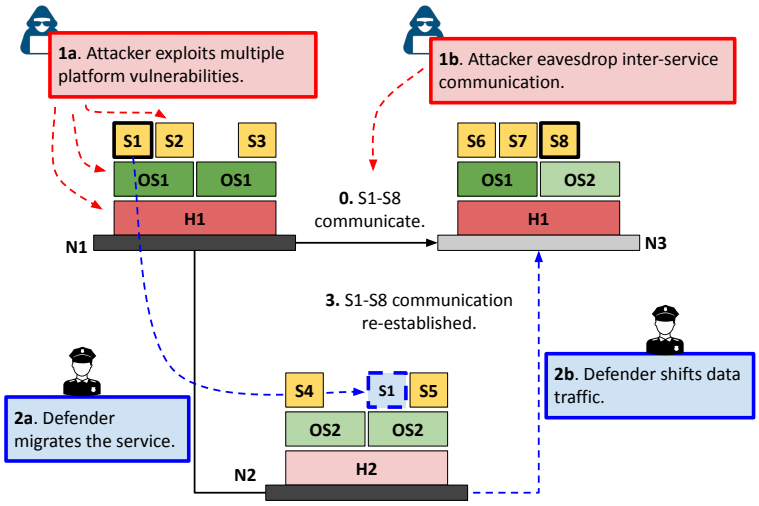

Abbildung 4: Ein Verteidigungsszenario mit beweglichem Ziel.

Strategien zu finden, die die potenzielle Kontrolle der Angreifer über das System minimieren. Diese Strategien legen fest, wie der Einsatz von Diensten rekonfiguriert und ihre Interkommunikation in einer dienstorientierten Architektur umgelenkt werden kann, um eine einfache Entdeckung des Systems zu verhindern und die Kontrolle über potenziell kompromittierte Anlagen wiederzuerlangen. Dies erfordert (i) die Frage, welche Dienstkonfiguration ein Verteidiger zu einem bestimmten Zeitpunkt verwenden sollte und (ii) wann der Verteidiger die Dienste gegen alle potenziellen Angriffsszenarien neu konfigurieren sollte. Wir haben unser früheres Modell JSAR (in Abschnitt 3.3) verwendet, um alternative Dienstkonfigurationen für (i) zu generieren, und dann ein neues Modell, PLSCH, entwickelt, das von einem bestehenden spieltheoretischen Rahmen inspiriert ist, um (ii) anzugehen. Die Kombination der beiden Modelle liefert schließlich eine umfassende Verteidigungsstrategie für die gegebenen Angriffsszenarien unter den Constraints begrenzter Systemressourcen und eines gegebenen Verteidigungsbudgets, z. B. zur Minimierung potenzieller Dienstunterbrechungen aufgrund von Rekonfigurationen. Die Ergebnisse dieser Arbeit zeigen die Herausforderungen bei der Entwicklung einer effektiven Strategie gegen alle potenziellen Bedrohungen. Die Einzelheiten dieser Modelle und ihre Bewertungsergebnisse wurden kürzlich bei einer internationalen Konferenz [14] eingereicht.

3.10 Prüfstand und Demonstration

Wir haben ein Testbed aufgebaut, um unsere Beiträge zu demonstrieren. In Abbildung 5 sind drei TSN-Switches in einer Ring-Topologie verbunden. Ein Server (als eingebettetes Modul, Raspberry Pi) ist mit dem ersten TSN-Switch (rechts) verbunden und die beiden anderen (Raspberry Pis) sind mit dem dritten Switch (links) verbunden. Sie sind virtualisiert, um isolierte kritische und nichtkritische virtuelle Domänen zu hosten und stellen DELIA-Module dar. Der DELIA-Supervisor und unser Monitoring-Tool, TSNZeek (siehe Abschnitt 3.8), ist mit dem zweiten Switch in der Mitte verbunden.

3.11 Weitere Projektaufgaben

Neben dem technischen Aufwand und den wissenschaftlichen Beiträgen leistete die UHH auch einen Beitrag zur Verwaltung, Wartung und Organisation des Projekts. Wir haben eine Mailingliste für die regelmäßige Kommunikation mit den Projektpartnern eingerichtet. Zur Unterstützung der Software-Entwicklung haben wir den Konsortialpartnern ein gitlab zur Verfügung gestell. Die UHH hat zudem

Abbildung 5: TSN-Testbed

Projekt-Meetings koordiniert und veranstaltet.

4 Publikationen und studentische Abschlussarbeiten

4.1 Publikationen

Im Folgenden führen wir alle zwölf Veröffentlichungen und zwei technische Reports auf, die im Rahmen von DELIA entstanden sind. Die Veröffentlichungen wurden ausschließlich auf internationalen Konferenzen und in Fachzeitschriften mit Peer-Review veröffentlicht.

[1] D. Ergenç, "DELIA Software Architecture," tech. rep., University of Hamburg, 2019.

[2] D. Ergenç, "A Modern Ethernet-based Arch. for Future Avionics," tech. rep., University of Hamburg, 2020.

[3] D. Ergenç and M. Fischer, "Resilience of Virtualized Embedded IoT Networks," *2. KuVS Fachgespräch "Network Softwarization"*, 2020.

[4] D. Ergenç, J. Rak, and M. Fischer, "Service-Based Resilience for Embedded IoT Networks," *IEEE/IFIP International Conf. on Dependable Systems and Networks (DSN)*, 2020.

[5] D. Ergenç, J. Rak, and M. Fischer, "Service-based Resilience via Shared Protection in Mission-critical Embedded Networks," *IEEE Transactions on Network and Service Management (TNSM), Special Issue on Design and Management of Reliable Communication Networks*, 2021.

[6] D. Ergenç, D. Sorejevic, and M. Fischer, "Distributed Bio-inspired Configuration of Virutalized Mission-critical Networks," *IEEE Global Communications Conference (GLOBECOM)*, 2022.

[7] N. Sertbaş Bülbül, D. Ergenç, and M. Fischer, "Evaluating Dynamic Path Reconfiguration for Time-sensitive Networks," *Würzburg Workshop on Next-Generation Communication Networks*, 2022.

[8] N. Sertbaş Bülbül, D. Ergenç, and M. Fischer, "SDN-based Self-Configuration for Time-Sensitive IoT Networks," *International Conference on Local Computer Networks (LCN)*, 2021.

[9] N. Sertbaş Bülbül, D. Ergenç, and M. Fischer, "Towards SDN-based Dynamic Path Reconfiguration for TSN," *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2022.

[10] D. Ergenç and M. Fischer, "On the Reliability of IEEE 802.1CB FRER," *IEEE International Conference on Computer Communications (INFOCOM)*, 2021.

[11] D. Ergenç and M. Fischer, "Implementation and Orchestration of IEEE 802.1CB FRER in OMNeT++," *IEEE International Conference on Communications (ICC), Workshop on Time-sensitive and Deterministic Networking*, 2021.

[12] D. Ergenç, C. Bruelhart, J. Neumann, L. Krueger, and M. Fischer, "On the Security of IEEE 802.1 Time-Sensitive Networking," *IEEE International Conference on Communications (ICC), Workshop on Time-sensitive and Deterministic Networking*, 2021.

[13] D. Ergenç, R. Schenderlein, and M. Fischer, "TSNZeek: An Open-source Intrusion Detection System for Time-sensitive Networking," *3rd International Workshop on Time-Sensitive and Deterministic Networking (TENSOR) at IFIP Networking*, 2023. submitted.

[14] D. Ergenç, F. Schneider, P. Kling, and M. Fischer, "Moving Target Defense for Service-oriented Mission-critical Networks," *IEEE 32nd International Conference on Computer Communications and Networks (ICCCN)*, 2023. submitted.

4.2 Studentische Abschlussarbeiten und Projekte

Die folgenden studentischen Arbeiten und Projekte, [15] [16], und [17] wurden im Kontext von DELIA durchgeführt. Es gibt noch laufende studentische Arbeiten zum Topologieentwurf und zur Erkennung von Cyber-Angriffen in zeitsensiblen und virtualisierten Netzen, die auf den DELIA-Ergebnissen aufbauen.

[15] D. Sorejevic, "Distributed Bio-inspired Configuration of Virtualized Mission-critical Networks," Masterarbeit, Universität Hamburg, 2022.

[16] R. Schenderlein, "Moving Target Defense for cloud-based Microservices," Bachelorarbeit, Universität Hamburg, 2022.

[17] P. C. Gomez, "Failover Profiling and Strategies for Mission-critical Embedded Networks," Studienarbeit, Universität Hamburg, 2021.

4.3 Open-Source-Beiträge

Wir haben im Rahmen des Projekts zwei Open-Source-Softwaremodule entwickelt[2].

Simulator für IEEE 802.1CB FRER und IEEE 802.1aq SPB Wir haben Erweiterungen für einen bestehenden Netzwerksimulator, OMNeT++, implementiert, um FRER und SPB zu simulieren. Dieser erweiterte Simulationsrahmen ermöglicht es uns, verschiedene Metriken für eine zuverlässige redundante Pfadkonfiguration für FRER im Rahmen unseres Beitrags zu bewerten, dessen Einzelheiten in Abschnitt 3.6 aufgeführt sind. Die Einzelheiten der Implementierung werden auch in einem anderen Beitrag [11] vorgestellt.

Überwachungs- und Einbruchserkennungssystem für IEEE 802.1 TSN-Protokolle Wir haben ein bestehendes Open-Source-Überwachungssystem, Zeek, erweitert, um zeitempfindliche Ethernet-Frames zu verarbeiten. Es ist in der Lage, den gesamten zeitempfindlichen Datenverkehr zu protokollieren und potenzielle Sicherheitsvorfälle in Bezug auf die IEEE 802.1Qcc SRP- und IEEE 802.1CB FRER-Protokolle, die für DELIA relevant sind, zu melden. Zum Zeitpunkt seines Entwurfs ist es das erste quelloffene Intrusion Detection System für IEEE 802.1 TSN. Wir haben auch ein Konferenzpapier [13] eingereicht, dessen Implementierungsdetails in Abschnitt 3.8 aufgeführt sind.

[2]Beide Module sind in unserem Universitätsrepositorium öffentlich zugänglich https://github.com/UHH-ISS/.

5 Impact

Die Arbeiten der UHH im Kontext von DELIA wurden in Form von 12 Veröffentlichungen (in Abschnitt 4.1) auf internationalen Konferenzen und Fachzeitschriften der Forschungsgemeinschaft präsentiert. Neben einer Verwertung der Ergebnisse ermöglichte dies auch die Einholung von wertvollem Feedback für den weiteren Projektverlauf. Mit insgesamt drei studentischen Abschlussarbeiten und Projekten haben wir Studierende in das Teilprojekt eingebunden. Zudem wir der über das Projekt beschäftigte Mitarbeiter, Doganalp Ergenc, noch in diesem Jahr seine Dissertationsschrift einreichen. Auf der Basis der Arbeiten in DELIA habe ich eine bestehende Vorlesung um ein Kapitel zu Time-sensitive Networking erweitert. Damit hatte und hat das Teilprojekt auch einigen Impact bei der Qualifikation des wissenschaftlichen Nachwuchs.
Des Weiteren hat das Teilprojekt zwei Open-Source-Beiträge hervorgebracht. Einer unserer Open-Source-Beiträge ist der erste öffentlich verfügbare Sicherheitsmechanismus für TSN-Netzwerktechnologien (Abschnitt 3.8 und 4.3). Wir haben die Ergebnisse mehrerer wissenschaftlicher Beiträge und unsere Softwaremodule genutzt, um eine Testumgebung zu entwickeln, in der wir verschiedene Anwendungsfälle für die wichtigsten DELIA-Funktionen, potenzielle Sicherheitsvorfälle und Systemausfälle demonstrieren können. Diese Testumgebung ermöglicht es uns auch, weitere Experimente zu den jeweiligen Themen durchzuführen. Vor allem aber zeigt es, dass die resultierende Software, die Optimierungsmodelle, die Fehlertoleranz und die Sicherheitsmechanismen in einem funktionierenden Prototyp eingesetzt werden können und somit vielversprechend sind, um für zukünftige missionskritische Systeme in Betracht gezogen zu werden, potenziell auch über die Avionik hinaus.
Schließlich hat DELIA auch dazu beigetragen, neue Forschungsprobleme zu identifizieren und zu Folgeprojekten geführt. Vor kurzem wurde mit RESISTANT ein neues Projekt im Luftfahrtforschungsprogramm des BMWK unter Beteiligung der Universitäten Hamburg und Stuttgart, als Mitglieder des DELIA-Konsortiums, gestartet. RESISTANT baut auf den Ergebnissen aus DELIA auf und legt einen stärkeren Fokus auf Resilienz von safety-kritischen und nicht safety-kritischen Systemen inklusive deren Monitoring auf Fehler und Cyber-Angriffe.

6 Zusammenfassung

Das Hauptziel von DELIA war die Erforschung der Entwurfsprinzipien für eine flexible und widerstandsfähige Mobilität der nächsten Generation, wobei der Schwerpunkt auf der Avionik lag. Dazu gehörte auch die Nutzung zweier neuer Entwurfsparadigmen, dienstorientierte Architekturen über Virtualisierung und Ethernet-basierte zeitsensitive Netze, um erweiterbare, offene und rekonfigurierbare Lösungen zu entwickeln. Die UHH war insbesondere für die Entwicklung von Fehlertoleranz- und Sicherheitsmechanismen zum Schutz solcher Systeme verantwortlich.
Die Hauptziele der UHH in DELIA waren dabei:

- Die Implementierung des DELIA-Software-Stacks, um eine flexible und serviceorientierte Architektur zu ermöglichen.
- Die Entwicklung zentraler und dezentraler Service-Orchestrierungsstrategien.
- die Entwicklung von Ausfallsicherheitsstrategien, z.B. für dynamische Failover und Rekonfiguration, unter Ausnutzung dieser Flexibilität.
- Die Erforschung der zuverlässigen Nutzung von zeitkritischen Netzwerkprotokollen.
- Die Entwicklung von Mechanismen zur Überwachung und Erkennung von Cyber-Angriffen so-

wohl für virtuelle Instanzen als auch für zeitkritische Kommunikation.

- Die Demonstration der effektiven Zusammenarbeit all dieser vorgeschlagenen Lösungen.

Wir haben erfolgreich zu all diesen Zielen beigetragen und im Kontext des Projekts 12 wissenschaftliche Artikel, zwei technische Berichte und zwei Open-Source-Beiträge veröffentlicht. Zur Demonstration der Projektergebnisse haben wir eine Testumgebung entwickelt, die es uns ermöglicht neue Forschungsideen zu realisieren und zu evaluieren. Darüber hinaus ist aus DELIA mit RESISTANT bereits ein Folgeprojekt entstanden, das Ende 2022 gestartet ist.

7 Bibliography

[18] J. Chenni Kumaran and M. Aramudhan, "A Survey on Resource Allocation Strategies in Cloud," *International Journal of Reasoning-based Intelligent Systems*, vol. 10, no. 3-4, pp. 328–336, 2018.

[19] N. K. Pandey, S. Chaudhary, and N. K. Joshi, "Resource Allocation Strategies used in Cloud Computing: A Critical Analysis," in *IEEE Conf. on Communication, Ctrl. and Intelligent Syst. (CCIS)*, 2017.

[20] D. Espling, L. Larsson, W. Li, J. Tordsson, and E. Elmroth, "Modeling and Placement of Cloud Services with Internal Structure," *IEEE Transactions on Cloud Computing*, vol. 4, no. 4, pp. 429–439, 2016.

[21] D. Breitgand, A. Marashini, and J. Tordsson, "Policy-driven service placement optimization in federated clouds," *IBM Research Division, Tech. Rep*, vol. 9, pp. 11–15, 2011.

[22] L. Pu, L. Jiao, X. Chen, L. Wang, Q. Xie, and J. Xu, "Online Resource Allocation, Content Placement and Request Routing for Cost-efficient Edge-caching in Cloud Radio Access Networks," in *IEEE Journal on Selected Areas in Communications*, vol. 36, pp. 1751–1767, 2018.

[23] M. B. Gawali and S. K. Shinde, "Task Scheduling and Resource Allocation in Cloud Computing using a Heuristic Approach," *Journal of Cloud Computing*, vol. 7, no. 1, 2018.

[24] X. Li and C. Qian, "A Survey of Network Function Placement," pp. 948–953, 2016.

[25] B. Yi, X. Wang, K. Li, S. k. Das, and M. Huang, "A Comprehensive Survey of Network Function Virtualization," *Computer Networks*, vol. 133, pp. 212–262, 2018.

[26] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual Network Functions Placement and Routing Optimization," in *IEEE Int. Conf. on Cloud Networking, CloudNet*, pp. 171–177, IEEE, 2015.

[27] M. F. Bari, S. R. Chowdhury, R. Ahmed, R. Boutaba, and O. C. M. B. Duarte, "Orchestrating Virtualized Network Functions," *IEEE Transactions on Network and Service Management*, vol. 13, no. 4, pp. 725–739, 2016.

[28] J. Liu, W. Lu, F. Zhou, P. Lu, and Z. Zhu, "On Dynamic Service Function Chain Deployment and Readjustment," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 543–553, 2017.

[29] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos, and L. P. Gaspary, "Piecing Together the NFV Provisioning Puzzle: Efficient Placement and Chaining of Virtual Network Functions," in *IFIP/IEEE Int. Symp. Integrated Netw. Mgmt. (IM)*, pp. 98–106, 2015.

[30] G. Lee, M. Kim, S. Choo, S. Pack, and Y. Kim, "Optimal Flow Distribution in Service Function Chaining," in *ACM International Conference Proceeding Series*, vol. 08-10-June-2015, pp. 17–20, 2015.

[31] P. Danielis, G. Dán, J. Gross, and A. Berger, "Dynamic flow migration for delay constrained traffic in software-defined networks," in *IEEE GLOBECOM*, pp. 1–7, IEEE, 2017.

[32] A. Alnajim, S. Salehi, and C.-C. Shen, "Incremental path-selection and scheduling for time-sensitive networks," in *IEEE GLOBECOM*, 2019.

[33] N. G. Nayak, F. Dürr, and K. Rothermel, "Incremental flow scheduling and routing in time-sensitive sdn," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 5, pp. 2066–2075, 2017.

[34] S. Kehrer, O. Kleineberg, and D. Heffernan, "A comparison of fault-tolerance concepts for IEEE 802.1 Time Sensitive Networks (TSN)," in *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, pp. 1–8, Sep. 2014.

[35] F. Prinz, M. Schoeffler, A. Lechler, and A. Verl, "End-to-end Redundancy between Real-time I4.0 Components based on Time-Sensitive Networking," in *IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, pp. 1083–1086, Sep. 2018.

[36] S. Qian, F. Luo, and J. Xu, "An Analysis of Frame Replication and Elimination for Time-Sensitive Networking," in *Proceedings of the 2017 VI International Conference on Network, Communication and Computing*, ICNCC 2017, (New York, NY, USA), p. 166–170, Association for Computing Machinery, 2017.

[37] G. Ditzel, "High Availability in EtherNet/IP Systems Using Frame Replication and Elimination for Reliability (FRER) as defined in the TSN Standard IEEE 802.1CB-2017," in *ODVA 18th Industry Conference and Annual Meeting*, October 2018.

[38] R. Hofmann, B. Nikolić, and R. Ernst, "Challenges and Limitations of IEEE 802.1CB-2017," *IEEE Embedded Systems Letters*, pp. 1–1, 2019.

[39] P. Meyer, T. Häckel, F. Korf, and T. C. Schmidt, "DoS Protection through Credit Based Metering - Simulation Based Evaluation for Time-Sensitive Networking in Cars," *CoRR*, vol. abs/1908.09646, 2019.

[40] P. Meyer, T. Häckel, F. Korf, and T. C. Schmidt, "Network Anomaly Detection in Cars based on Time-Sensitive Ingress Control," in *IEEE 92nd Vehicular Technology Conference*, 2020.

[41] P. Meyer, T. Häckel, S. Reider, F. Korf, and T. C. Schmidt, "Network Anomaly Detection in Cars: A Case for Time-Sensitive Stream Filtering and Policing," *CoRR*, vol. abs/2112.11109, 2021.

[42] IEEE 802.1 TSN Task Group, "IEEE Std. for Local and Metro. Area Net. – Bridges and Bridged Networks – Amendment 28: Per-Stream Filtering and Policing," *IEEE Std 802.1Qci-2017*, 2017.

[43] F. Luo, B. Wang, Z. Fang, Z. Yang, Y. Jiang, and K. Demertzis, "Security Analysis of the TSN Backbone Arch. and Anomaly Det. System Design Based on IEEE 802.1Qci," *Sec. and Comm. Networks*, 2021.

[44] R. Barton, M. Seewald, and J. Henry, "Management of IEEE 802.1Qci Security Policies for Time Sensitive Networks (TSN)," tech. rep., Technical Disclosure Commons, 10 2018.

[45] T. Bu, Y. Yang, X. Yang, W. Quan, and Z. Sun, "TSN-Insight: An Efficient Network Monitor for TSN Networks," APNet, 2019.

[46] L. Muguira, J. Lázaro, S. Alonso, A. Astarloa, and M. Rodriguez, "Secure Critical Traffic of the Electric Sector over Time-Sensitive Networking," in *IEEE 35th Conference on Design of Circuits and Integrated Systems (DCIS)*, 2020.

[47] IEEE 802.1 TSN Task Group, "IEEE Std. for Local and Metro. Area Net. – Media Access Control (MAC) Security," *IEEE Std. 802.1AE-2018*, 2018.

[48] *Automotive Virtual Platform Specification.* GENIVI, Jul 2020.

[49] *Future Airborne Capability Environment (FACE) Technical Standard.* The Open Group, Jul 2020.

[50] *IMMUNE: Selbstverteidigende Netzwerk für resiliente Industrie 4.0.*

[51] *secVI: Security for Vehicular Information.* BMBF, 2019. https://secvi.inet.haw-hamburg.de/.

[52] *AIDA: A Hollistic AI-driven Networking and Processing Framework for Industrial IoT.* KK-stiftelsen, 2020. https://sola.kau.se/aida/.

[53] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[54] A. Shostack, S. Hernan, S. Lambert, and T. Ostwald, "Uncover Security Design Flaws Using The STRIDE Approach."

DELIA - Distributed, Extendable, Lightweight, open, reliable, servIce-oriented Architecture for next-gen mobility

Kurzbericht des Teilprojekts der Universität Hamburg, Förderkennzeichen 16KIS0941
Förderprogramm KMU-innovativ: Informations- und Kommunikationstechnologie (2019-2022)
Doğanalp Ergenç und Prof. Dr. Mathias Fischer

1 Kurzbeschreibung

Moderne missionskritische Systeme, z. B. wie sie in der Automobil- und Avionikbranche zu finden sind, sind heterogene und komplexe Ökosysteme. Sie bestehen aus einer Vielzahl von miteinander verbundenen Komponenten und internen Funktionen und zeichnen sich durch eine große Vielfalt unterschiedlicher Hardware und Software aus. Diese alles erschwert die Entwicklung solcher Systeme und schränkt ihre Erweiterbarkeit und Rekonfigurierbarkeit ein. Daher werden mittlerweile verstärkt neue Technologien, wie Virtualisierung, Service-oriented Architectures (SOA) oder Time-sensitive Networking (TSN) in missionskritischen Systemen angewendet, um ihre Entwicklung zu erleichtern. Die Virtualisierung eingebetteter Komponenten ermöglicht die Anwendung von SOA für sicherheitskritische Systeme. Kritische Systemdienste können so flexibel und virtualisiert auf General-Purpose-Hardware realisiert werden, anstatt auf dedizierter Hardware mit begrenzten Fähigkeiten zu laufen. In ähnlicher Weise helfen die jüngsten IEEE-Standards für zeitkritische Netzwerke (TSN) dabei, deterministische Kommunikation zwischen gemischt-kritischen Diensten und mittels handelsüblicher und daher günstiger Hardware zu realisieren. Diese neuen Technologien ermöglichen zudem die Verwendung neuer Methoden für die Erhöhung von Sicherheit und Zuverlässigkeit von missionskritischen Systemen insbesondere auch gegenüber Fehlern und Cyber-Angriffen.
Das Projekt DELIA und insbesondere auch das Teilprojekt der Universität Hamburg zielte darauf ab, eine skalierbare, leichtgewichtige, erweiterbare und resiliente Kommunikations- und Verarbeitungsplattform auf der Basis von SOA und TSN zu entwickeln. Die UHH hat sich dazu vor allem auf die Entwicklung der erforderlichen Software-Module fokussiert die auf DELIA-Hardware-Modulen lauffähig sind. DELIA-Instanzen sollen in der Lage sein, kritische und nicht kritische virtuelle Dienste isoliert zu hosten und miteinander zu kommunizieren, um komplexe Aufgaben kritischer Netze zu bewältigen, wobei der Schwerpunkt auf Avioniksystemen liegt.

2 Verlauf des Projekts

Die Universität Hamburg (UHH) hat im Projekt wesentliche Beiträge zur DELIA-Plattform geleistet. Dies umfasste die Entwicklung sicherer, geschützter System-Software-Funktionen sowie von Sicherheits- und Resilienzmaßnahmen inklusive der Umsetzung von Methoden für sichere, zeitkritische Kommunikation zwischen den DELIA-Knoten. Im Detail hat die UHH die folgenden Arbeiten umgesetzt:

1. Die UHH war für die Koordination der Entwicklung der Systemarchitektur verantwortlich, mit einem Fokus auf der Sicherheit der Dienstverteilung und der Kommunikation zwischen den Diensten. Des Weiteren wurden Sicherheitsanforderungen an die Kommunikations- und Prozessorplattform und ein Angreifermodel definiert sowie eine Analyse der vorhandenen Software und der besten Sicherheitsverfahren für eine solche Plattform vorgenommen.

2. Die UHH war für die Entwicklung einer Software verantwortlich, die die Verteilung und Verwaltung von Diensten ermöglicht und Sicherheitsfunktionen für das Gesamtsystem umsetzt. Dazu gehörte auch die Umsetzung von Maßnahmen zur resilienten, zeitsensitiven Vernetzung von

DELIA-Knoten sowie die Implementierung von Werkzeugen für die Wartung und des Monitorings der DELIA-Plattform.

3. Die UHH hat die entwickelte Software auf der DELIA-Hardware lauffähig gemacht und weitere Mechanismen für verbesserte IT-Sicherheit und Resilienz integriert und diese hinsichtlich Fehlertoleranz und gegenüber Cyber-Angriffen evaluiert.

3 Beiträge und Ergebnisse

Die UHH hat alle Ziele des Teilprojekts erreicht. Im Projektverlauf hat die UHH 12 wissenschaftliche und projektbezogene Artikel auf Konferenzen und in Journals sowie zwei technische Berichte veröffentlicht. Zudem wurden zwei Open-Source-Beiträge veröffentlicht. Nachfolgend beschreiben wir kurz die wichtigsten Beiträge und Ergebnisse:

- Gemeinsam mit der Universität Stuttgart haben wir geeignete Virtualisierungstechnologien, Betriebssysteme und Tools für DELIA-Knoten identifiziert. Nachfolgend haben wir die notwendigen Software-Module implementiert, um gemischt-kritische miteinander kommunizierende und virtualisierte Dienste in eine zugrundeliegende missionskritische Hardware-Plattform einzubetten.

- Wir haben den DELIA-Supervisor implementiert, der auf der DELIA-Plattform laufende virtuelle Dienste orchestriert und überwacht sowie im Falle eines Ausfalls oder Angriffs automatisch repariert. Darüber hinaus haben wir verschiedene Optimierungsmodelle für die Bereitstellung von Diensten und die Kommunikation zwischen den Diensten vorgeschlagen, die sich zentralisiert auf dem DELIA-Supervisor ausführen und lösen lassen. Schließlich haben wir verteilte, bio-inspirierte Methoden zur (Selbst-)Konfiguration von Diensten und Kommunikation vorgeschlagen, um die Abhängigkeit von einem einzigen zentralen Controller zu verringern.

- Die UHH hat TSN-Protokolle für die Erhöhung der Fehlertoleranz untersucht, Probleme identifiziert und Verbesserungen vorgeschlagen, die diese Protokolle auch auf komplexen Netztopologien sicher einsetzbar machen. Im Zuge der Analyse wurde ein Simulationsmodell zur Bewertung dieser Protokolle entwickelt, dass als Open-Source freigegeben wurde.

- Wir haben zusammen mit dem Partner ZAL TSN-Protokolle auf potenzielle Sicherheitsbedrohungen analysiert und dabei mehr als 30 mögliche Angriffe auf TSN identifiziert. Unter Berücksichtigung dieser potenziellen Sicherheitsprobleme haben wir mit TSN-Zeek ein System zur Erkennung von Angriffen auf TSN-Protokolle entwickelt und dieses auch als Open-Source freigegeben.

- Schließlich hat die UHH ein eigenes Testbed aufgebaut, um alle Beiträge zu integrieren und in Beispielszenarien zu demonstrieren. Darüber konnte gezeigt werden, dass die von der UHH entwickelte Software die angriffs- und fehlertolerante Einbettung von virtualisierten Dienstetopologien inklusive resilienter Netzwerkkommunikation unter Wahrung von Quality-of-Service-Anforderungen ermöglicht.