TECHNISCHE
UNIVERSITÄT
DARMSTADT

PRIVACY-AWARE and RELIABLE COMPLEX EVENT PROCESSING in the INTERNET of THINGS

Trust-Based and Flexible Execution of Event Processing Operators in Dynamic Distributed Environments

Vom Fachbereich Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt
zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Dissertation

von

RAHUL WERMUND, GEB. CHINI DWARAKANATH, M.SC.

Geboren am 05. SEPTEMBER 1987 in Raichur, Indien

Vorsitz: Prof. Dr.-Ing. Jürgen Adamy
Referent: Prof. Dr.-Ing. Ralf Steinmetz
Korreferent: Prof. Dr.-Ing. Bernd Freisleben

Tag der Einreichung: 30. Oktober 2017
Tag der Disputation: 21. Dezember 2017

Hochschulkennziffer D17
Darmstadt 2018

Dieses Dokument wird bereitgestellt von tuprints, E-Publishing-Service der Technischen Universität Darmstadt.
http://tuprints.ulb.tu-darmstadt.de
tuprints@ulb.tu-darmstadt.de

This document is provided by

Bitte zitieren Sie dieses Dokument als:
URN: urn:nbn:de:tuda-tuprints-72138
URL: http://tuprints.ulb.tu-darmstadt.de/id/eprint/7213

Please cite this document as:

Die Veröffentlichung steht unter folgender Creative Commons Lizenz:
Namensnennung - Nicht kommerziell - Keine Bearbeitungen 4.0 International
https://creativecommons.org/licenses/by-nc-nd/4.0/deed.de

This publication is licensed under the following Creative Commons License:
Attribution-NonCommercial-NoDerivatives 4.0 International
https://creativecommons.org/licenses/by-nc-nd/4.0/deed.en

ABSTRACT

The Internet of Things (IoT) promises to be an enhanced platform for supporting a heterogeneous range of context-aware applications in the fields of traffic monitoring, healthcare, and home automation, to name a few. The essence of the IoT is in the inter-networking of distributed information sources and the analysis of their data to understand the interactions between the physical objects, their users, and their environment. Complex Event Processing (CEP) is a cogent paradigm to infer higher-level information from atomic event streams (e.g., sensor data in the IoT). Using functional computing modules called *operators* (e.g., filters, aggregates, sequencers), CEP provides for an efficient and low-latency processing environment.

Privacy and mobility support for context processing is gaining immense importance in the age of the IoT. However, new mobile communication paradigms—like Device-to-Device (D2D) communication—that are inherent to the IoT, must be enhanced to support a privacy-aware and reliable execution of CEP operators on mobile devices. It is crucial to preserve the differing privacy constraints of mobile users, while allowing for flexible and collaborative processing. Distributed mobile environments are also susceptible to adversary attacks, given the lack of sufficient control over the processing environment. Lastly, ensuring reliable and accurate CEP becomes a serious challenge due to the resource-constrained and dynamic nature of the IoT.

In this thesis, we design and implement a privacy-aware and reliable CEP system that supports distributed processing of context data, by flexibly adapting to the dynamic conditions of a D2D environment. To this end, the main contributions, which form the key components of the proposed system, are three-fold:

(i) We develop a method to analyze the communication characteristics of the users and derive the type and strength of their relationships. By doing so, we utilize the behavioral aspects of user relationships to automatically derive differing privacy constraints of the individual users.

(ii) We employ the derived privacy constraints as trust relations between users to execute CEP operators on mobile devices in a privacy-aware manner. In turn, we develop a trust management model called TrustCEP that incorporates a robust trust recommendation scheme to prevent adversary attacks and allow for trust evolution.

(iii) Finally, to account for reliability, we propose FlexCEP, a fine-grained flexible approach for CEP operator migration, such that the CEP system adapts to the dynamic nature of the environment. By extracting intermediate operator state and by leveraging device mobility and instantaneous characteristics, FlexCEP provides a flexible CEP execution model under varying network conditions.

Overall, with the help of thorough evaluations of the above three contributions, we show how the proposed distributed CEP system can satisfy the requirements established above for a privacy-aware and reliable IoT environment.

KURZFASSUNG

Das Internet der Dinge (engl. *Internet of Things* [IoT]) wird künftig eine verbesserte Plattform für heterogene kontextbewusste Anwendungen unter anderem in den Bereichen von Verkehrsbeobachtung, Gesundheitsfürsorge und Heimautomation bieten. Das Hauptprinzip des IoT ist die Vernetzung von verteilten Informationsquellen (z.B. Sensoren in der Umgebung) und die Analyse ihrer Daten, um Einblicke in die Zusammenhänge zwischen den physischen Entitäten, deren Nutzern, und deren Umgebung zu gewinnen. Komplexe Ereignisverarbeitung (engl. *Complex Event Processing* [CEP]) bietet eine stichhaltige Lösung, höherwertige Informationen aus den atomaren Ereignisströmen (bspw. Sensordaten im IoT) abzuleiten. Durch funktionale Rechenmodule, sogenannte *Operatoren* (z.B. Filter, Aggregator, Sequenzer), verschafft CEP eine effiziente Verarbeitungsumgebung mit niedriger Latenz.

Die Sicherstellung der Privatsphäre und die Unterstützung der Mobilität von Nutzern ist im Zeitalter des IoT von enormer Bedeutung. Doch neue mobile Kommunikationsparadigmen im IoT, wie Gerät-zu-Gerät (engl. *Device-to-Device* [D2D]) Kommunikation, müssen erweitert werden, um eine privatsphärenbewusste und zuverlässige Ausführung von CEP auf mobilen Geräten zu unterstützen. Für mobile Nutzer müssen dabei vielfältige und unterschiedliche Bedingungen an die Privatsphäre sichergestellt werden. Gleichzeitig muss das CEP-System ein hohes Maß an Flexibilität sowie Kollaboration zwischen den Nutzern unterstützen, um relevante Informationen für die Nutzer erkennen zu können. Verteilte Umgebungen können außerdem aufgrund der fehlenden Kontrolle über die einzelnen Systemkomponenten Angriffen ausgesetzt sein. Schließlich stellt die Aufrechterhaltung eines zuverlässigen und akkuraten CEP-Systems aufgrund der Ressourceneinschränkungen sowie Dynamik des IoT eine erhebliche Herausforderung dar.

In dieser Dissertation wird ein privatsphärenbewusstes und zuverlässiges CEP-System entwickelt, implementiert und evaluiert. Das CEP-System unterstützt eine verteilte Verarbeitung von Kontextdaten im IoT, indem es sich flexibel den dynamischen Rahmenbedingungen einer D2D-Umgebung anpasst. Die Hauptbeiträge dieser Arbeit, die auch die drei Hauptkomponenten des entwickelten Systems bilden, lauten wie folgt:

(i) Als erster Beitrag wird eine Methode zur Analyse der Kommunikation zwischen Nutzern entwickelt, um daraus die Art und Stärke der Beziehungen abzuleiten. Somit werden die verhaltensbezogenen Eigenschaften von Nutzerbeziehungen in Anspruch genommen, um automatisch die unterschiedlichen Privatsphärebedingungen der einzelnen Nutzer abzuleiten;

(ii) Im zweiten Beitrag werden die abgeleiteten Privatsphärebedingungen zum Ermitteln des Vertrauens zwischen Nutzern angewendet, um somit die CEP-Operatoren unter Sicherstellung der Privatsphäre auf mobilen Geräten umsetzen zu können. Dazu wird ein Modell zur Verwaltung des Nutzervertrauens

namens TrustCEP konzipiert. Das Modell ermöglicht bei Angriffen anderer Nutzer robuste Empfehlungen hinsichtlich des aktuellen Nutzervertrauens, um dabei auch wechselndes Benutzerverhalten und Änderungen der Vertrauensbeziehungen über die Zeit zu berücksichtigen;

(iii) Im dritten Beitrag wird zur Unterstützung der Zuverlässigkeit ein feingranularer und flexibler Ansatz namens FlexCEP für die Migration von Operatoren entwickelt. Dieser erlaubt dem CEP-System, sich der Dynamik im Netzwerk anpassen zu können. FlexCEP bietet eine flexible Ausführung von CEP bei wechselnden Netzwerkbedingungen, indem einerseits der Zwischenzustand der Operatoren und andererseits die aktuellen Mobilitäts- und Leistungswerte der Geräte berücksichtigt werden.

Insgesamt wird durch umfassende Evaluationen gezeigt, dass das entwickelte verteilte CEP-System die oben beschriebenen Anforderungen für eine privatsphärenbewusste und zuverlässige IoT-Umgebung erfüllt.

ACKNOWLEDGMENTS

I HAVE liked numbers all through my life, and I find it fitting to describe my PhD life at KOM in numbers, as well—1438 days; 76 colleagues; 339934 characters in published papers; and 67 cups of coffee/cocoa (I probably hold the record for the *least* coffee/cocoa intake over a period of almost 4 years!).

For all these numbers, my sincere thanks go to all the 76 colleagues I have had the pleasure of working with during my time at KOM. . . starting, of course, with "*der Chef*" Ralf Steinmetz, who has created a unique working environment at KOM that fosters interdisciplinary collaboration and interaction among many talented people. Using this foundation, I have not only grown wiser, but also developed my social and technical skills substantially.

This PhD would not have been possible without the supervision and insightful observations of Boris, my group head over the past three years as well as CEP companion. His constant motivation and many fruitful (yet sometimes quite long and digressing ☺) discussions allowed me to formulate the research questions appropriately and answer them adequately. His feedback and critical, yet constructive evaluation of my work helped in improving the quality of the thesis considerably. A big thank you to my first group head, Dominik, who started it all and helped me get on track during the rather difficult, initial days of my PhD life. Also, I would like to express my gratitude to Prof. Bernd Freisleben for being my co-advisor and making me believe—even during a short 20-minute meeting—that I had done something worthwhile during my PhD career.

Coming to my group colleagues as part of AOC—the best group at KOM 😎—and I have had quite a few over the course of my PhD. Going alphabetically, huge thanks to Alex, Amr (thanks for the useful insights as the *other* group head), Björn, Christian (the cook), Denny (thanks for introducing me to *Cards against Humanity*! 😄), Jeremias, Leo, Manisha (hope our paper gets accepted!), Melanie, Nils, Rhaban (Bayern doof; heja BVB!), Ronny, Sonja, Sounak, and Stefan (not AOC, but still part of *our* group).

Work at KOM goes beyond one's own group, and in my case, I had a lot of interaction with the other group members. I'd like to thank all researchers for their company, advice (both motivational as well as technical), and support. In particular, special thanks to (again alphabetically) Binh, Doreen, Frank and Irina (**the** KN2 team!), Patrick, Sebastian (der ausm Schwabenländle), Steffen, Svenja (danke, dass du immer ein offenes Ohr für mich hattest!), Timmy (the fist-bump guy), both Tobis, and Wael (the billiards pal). All the best to those who aren't across the PhD hurdle, yet. Hope we all stay in touch in the future!

Two other groups of people are indispensable when it comes to completing a PhD at KOM—the students and the ATMs, i.e., the supporting staff. During these years at KOM, I had the privilege to supervise some talented students who aided the progress of my thesis considerably. In particular, I'd like to thank Jérôme—

who is now on course to become a pilot!—and Yashas—who landed a job at SAP thanks to his work on his Master thesis under my supervision; I couldn't be more proud. The ATMs are the backbone of KOM, and my thesis. Again going alphabetically, Britta, Frank (Jöst), Mrs. Scholz-Schmidt, Jan (danke für die Hilfe beim Führerschein!), Karola, Mrs. Ehlhardt, Moni, Sabine, as well as our *heavy metal* PR department, Matthias and Thomas (`\m/`). I could not thank you all enough for your help and support.

Of course, I had a life outside KOM, as well. 😄 After all, I also worked on the project SoLin that focused on achieving the optimal work-life balance for knowledge workers, like me. I would like to thank my project partners at MuP and Uni Kassel for their support and cooperation over the past four years, with special thanks to Arno, Christoph, Gisela, Kathi, Kathrin, Michael, Olga, and Sebastian (Wojtek).

Outside of work, huge thanks to my friends[1]—Ananth (thanks for hosting me in SF!), Carlos, Erin, Lars, Malte, Outi, Philipp, Sandy, Sebl (flat mate/band mate/-counselor), Shruthi, Sindhu, Suzi, Vibha, and the original Darmstädters, BJ, Guru, Naga, Dr.-Ing. Shabby, and VJ—who helped me keep a sane mind, whether it was by performing a gig somewhere, going out for sushi, playing board games on the weekends, or just having the occasional chat on WhatsApp/Skype.

Now for the *last-but-not-leasts*: My family has given me immense support all through my life, and especially during my PhD. I am so grateful to have the understanding and love from my parents and my brother, Rohit (who played football with us in KWT 2017!). To be honest, my dad is probably more proud of my PhD than I am! My extreme gratitude also goes to my parents-in-law, well, actually, they are like parents to me.

And finally, my best friend and now wife, Sabrina, thank you so much for everything! I know that the past couple of years, and especially, the past couple of months have been quite strenuous to the both of us, to say the least. I could not be more grateful to you for all the love, support, and encouragement you have given me. I love you.

Darmstadt, 2018 *Rahul Wermund, ne Chini Dwarakanath*

1 Apart from those at work who also became good friends

CONTENTS

1 INTRODUCTION

Recent years have seen a dramatic paradigm shift towards networked sensor-fitted devices (e.g., smartphones), and subsequently, the rise of context- and situation-aware applications. Such applications interact with users and their environment to provide specific services based on the prevailing user context. Examples can be found in the fields of environmental monitoring—observing noise and air pollution levels in the environment and marking the areas that users should avoid [57]—and traffic management—observing traffic movement and congestion levels to optimize traffic flow on highways [77]—as well as healthcare monitoring [103, 135] and home automation [138], to name a few.

A promising emerging concept that lays an improved platform for heterogeneous context-aware applications is the *Internet of Things* (IoT). The IoT entails the internetworking of mobile devices like smartphones, vehicles, and environmental sensors present in everyday infrastructure like buildings and electronics [31, 76, 203, 237]. Thus, the IoT offers the potential for an increased availability of information about the environment, leading to the notion of smart cities and smart infrastructure [203, 237].

A key concept in the IoT is *Device-to-Device* (D2D) communication, which is also set to be part of the *Fifth Generation* (5G) standard for the future cellular network architecture and future wireless systems, especially for improving network coverage [87, 211]. Given the resource-constrained nature of the devices involved in the IoT, D2D communication ensures better efficiency, especially with respect to communication costs and energy consumption in a mobile environment. Owing to direct communication among the devices and reduced dependency on a central entity, D2D communication allows for improved availability and accessibility of the sensing and processing devices [31, 140]. In this thesis, we mainly focus on *local* D2D environments, where devices process and exchange information collaboratively, without the necessity of a central entity.

Context processing and recognition is the most basic requirement for any context-aware application. One of the most promising paradigms for context processing is *Complex Event Processing* (CEP) or just *event processing*. CEP enables an efficient inference of higher-level data by discovering inherent patterns in event streams (e.g., sensor data) [36, 53, 54, 69]. Basically, CEP facilitates the interpretation of incoming event streams using computing modules, often called *operators*, and produces higher-level information with high accuracy and low latency [53, 157].

A major aspect of context processing in the IoT is data privacy, primarily in view of the possibility of the inference and processing of sensitive information about a user's location, co-location, activity, or mood [129, 184, 193]. Thus, ensuring the privacy of events in CEP systems poses a serious challenge, especially given the dis-

tributed nature of the IoT, where sensitive information can be processed on several (possibly malicious) devices [59, 61]. In addition to privacy, ensuring *reliability* during distributed event processing also poses a significant challenge in the IoT. Given the resource-constrained and dynamic nature of a D2D environment, any distributed processing system must adapt itself to the varying conditions of the environment.

In this thesis, we propose a distributed CEP system that facilitates the preservation of privacy as well as increases reliability in dynamic D2D environments in the IoT. In the following, we motivate our approach by highlighting some important concepts behind CEP as well as the main challenges towards the above-mentioned goal. We then draw the main research questions and corresponding contributions of this thesis.

1.1 MOTIVATION

The first event processing systems in related literature focused primarily on the filtering and extraction of relevant events from incoming notifications. This formed the basis for the well-known, message-oriented interaction paradigm called *publish/-subscribe* [40, 70, 165], where users can obtain information of interest by subscribing to particular topics or classes of events. CEP extends traditional publish/subscribe systems by allowing users to perform certain operations on *simple* events (e.g., temperature data) to obtain *complex* events (e.g., fire) [36, 53, 54].

Typically in CEP, multiple dependent operators, such as filters, aggregates, and sequencers, are used to produce complex events. The operators can be placed freely on processing elements, while the execution order and the corresponding input/output relationship of event streams are dictated by directed, acyclic graphs called *operator graphs*. These operator graphs connect *producers* (i.e., sources of information) to *consumers* (i.e., stakeholders of higher-level information) through event streams [54, 157]. By means of appropriate collaboration among available devices exchanging information, distributed CEP provides for an efficient and low-latency processing platform.

The main objective of any distributed CEP system is the adequate placement of the CEP operators on the available devices. A wide spectrum of methods have been proposed in related literature for the appropriate placement of CEP operators to satisfy different processing requirements [38, 54, 121, 157, 201]. CEP placement algorithms address an extensive range of issues, e.g., improving bandwidth and memory usage [23], minimizing processing and communication delays [54, 157, 159], and improving energy efficiency [201, 231]. Nevertheless, the execution of placement algorithms in a highly dynamic environment like D2D-based networks in the IoT imposes major difficulties with respect to the privacy and dynamic nature of both, the users and the devices involved. In the following, we detail these challenges to highlight the main requirements of the privacy-aware and reliable CEP system proposed in this thesis.

1.1.1 *Main Challenges*

Respecting the differing privacy concerns of users. Distributed processing in the IoT involves interaction and collaboration among users with possibly different privacy constraints. The privacy constraints depend strongly on the type of information shared as well as the intended recipient(s) [114, 153, 170]. For example, in a typical context recognition system, users may be willing to share their location readings with anyone by classifying them as non-private information. On the other hand, they may only share their microphone readings with close friends/colleagues, classifying them as extremely sensitive information. These constraints may vary from user to user, depending also on the time of the day and nature of the situation at hand. It is important to consider these privacy constraints while deploying CEP on available devices, such that user privacy constraints are protected while still facilitating flexible collaboration among devices.

Preventing adversary attacks. Distributed systems are generally susceptible to different kinds of adversary attacks. A fundamental issue in distributed processing is the difficulty to comprehensively track the path of events after they have been disseminated, and to control the activity of the other devices involved [195]. This becomes increasingly challenging in a D2D-based network, where the devices are mobile and interconnected intermittently. Adversaries may collaborate with other adversaries to obtain more sensitive information from a combination of received event streams. Preventing adversary attacks, hence, becomes a crucial factor for the proper functioning of one such system.

Adapting to a dynamic user environment. The placement and execution of CEP operator graphs varies with dynamic changes in the available sensor data as well as the available processing devices. The main problem with distributed CEP in D2D-based networks is possible network discontinuity due to device movement or failure [60, 110, 122, 159], which in turn leads to the loss of events and/or falsified output events. Accounting for device mobility is an important requirement in the IoT. The performance of devices like user smartphones can fluctuate due to the restricted amount of resources like battery life and memory space. Hence, it becomes pivotal to adapt the execution of CEP operator graphs in a flexible manner to the dynamic changes in the environment.

1.1.2 *Key Research Gaps*

There exists a large collection of approaches towards distributed CEP in related literature, covering a wide spectrum of system requirements. Most existing approaches focus on fixed networks and on performance metrics, such as latency, bandwidth, and network load [23, 54, 121, 167], by placing the CEP operators on appropriate devices. Other CEP-related approaches in the field of wireless and mobile networks concentrate on achieving either low latency event processing [157, 159] or energy-efficiency among resource-constrained devices [201, 231]. But these approaches neglect privacy altogether. Current CEP systems that do consider privacy as a constraint concen-

trate on access policy mechanisms to facilitate the processing of sensitive events in a distributed environment [91, 187]. However, these systems have been construed for static networks and scenarios with low dynamic behavior.

The flow of events from one device to another must take place in adherence to the privacy constraints of the users involved, which thereby has a direct correspondence with the placement of CEP operators. The privacy-aware placement of CEP operators becomes a crucial issue, especially when *a priori* knowledge of the sensing and processing devices is unavailable. Many related approaches in other distributed environments—e.g., *Peer-to-Peer* (P2P) networks, *Mobile Ad-hoc NETworks* (MANETs), or *Delay-Tolerant Networks* (DTN)—propose to tackle this issue by capitalizing on the *trust* between the devices to facilitate privacy-aware data processing [47, 124, 174, 206, 208, 216, 229]. However, these approaches either consider a trusted third party to allow for privacy-aware transactions, suffer from high computational delays, or do not devise appropriate methods to deal with the evolution of trust over time.

The constantly changing user landscape enforces additional restrictions on the operation of a distributed CEP system. Even after operator placement, appropriate measures must be taken to adapt system functionality to the dynamic changes. The migration of operators to less failure-prone devices can support a reliable execution of CEP operator graphs in these environments. While existing work on reliability in CEP systems has proposed cogent solutions for static and fixed networks [22, 65, 93, 116, 159, 188], there is a clear lack of approaches for the dynamic mobile environment, where resource constraints and device mobility are the norm.

1.2 RESEARCH QUESTIONS AND MAIN CONTRIBUTIONS

The main goal of our work is to design, realize, and evaluate a **privacy-aware and reliable CEP system** that allows for the **distributed processing** of context data, by flexibly adapting to the dynamic conditions of a D2D environment. Consequently, based on this goal and the analysis of the key research gaps, we address the following three research questions. For each of the research questions, we provide a brief description of the corresponding contributions in this thesis.

RQ1.1: *How can we automatically estimate user relationships in order to identify user privacy constraints?*

One of the primary factors that affects users' privacy constraints—especially with respect to the amount of information they would like to disclose or share—is their relationship with other users and the corresponding trust level [51, 114, 153]. In order to identify these privacy constraints, we investigate how we can estimate the type (social circle) and closeness (tie strength) of user relationships based on their interactions. In particular, we propose a method to analyze users' communication behavior on synchronous and asynchronous channels with the help of our smartphone-based application called FAMAPP for field studies, expanding our previous work [58]. In doing so, we determine the factors—e.g., intensity, frequency, locative dependence, and temporal dependence—that have the most influence on estimating user relationships.

RQ1.2: *What are the implications of user privacy constraints on CEP operator placement and execution?*

We leverage the concept of trust in supply chain management such that the trust level between two users (also termed *dyadic trust* [181]) dictates the amount of information shared between them [64]. We use the concept of *behavioral trust* [5, 190, 212] to capture the *direct* trust between the users, based on the analysis of their communication behavior, as per our contribution to RQ1.1. In doing so, we capitalize on the social networking characteristics of the user environment to establish a privacy-aware setting for event processing. We propose a trust-based approach called TrustCEP for distributed CEP [59], wherein we (i) propose a trust management model based on user communication behavior, (ii) incorporate a robust trust recommendation scheme to overcome adversary attacks and support trust evolution, and (iii) develop an algorithm to apply local trust for privacy-aware operator placement and execution of CEP operator graphs.

RQ1.3: *How can we ensure flexible execution of CEP operators in dynamic D2D environments?*

To overcome the challenges of reliability and efficient execution of CEP operator graphs in dynamic D2D environments, we propose FlexCEP, a flexible approach for fine-grained mobility-aware operator migration from one device to another in D2D-based networks in the IoT. To this end, building on previous work [60], we (i) demonstrate how operator state can be extracted from a standard CEP environment, and (ii) leverage device mobility and other instantaneous characteristics (e.g., current energy levels) to migrate partially extracted operator state in a resource-efficient manner. In turn, FlexCEP offers an increased flexibility in failure-prone environments, by trading between communication overhead and recovery time, depending on the application needs.

With these contributions, we satisfy the requirements for a privacy-aware and reliable CEP system in dynamic D2D environments. While our focus lies in the *local* D2D environment with direct interaction between devices, the above approaches can also be applied in other distributed environments on a *global* scale using concepts like fog computing [34, 35, 94, 157] and cloudlets [185, 215] that allow for hybrid solutions.

Furthermore, with respect to privacy, we mainly propose a trust-based approach that adhere to user privacy constraints in order to facilitate distributed processing. However, we assume that appropriate security mechanisms underlie our implementation, based on existing advancements in encryption and authentication [68, 97, 155, 162].

Finally, we assume the availability of an appropriate CEP query language and rule-set for different IoT applications. In turn, we assume that the generated CEP operator graphs can be executed in a distributed environment with many devices, as has been shown by many related efforts so far [54, 157, 201]. Thus, we do not focus on the generation of rule models and their semantics. Interested readers may refer to related work by Chakravarthy and Mishra [44], and by Cugola and Margara [52, 54].

1.3 THESIS STRUCTURE

In the following, Chapter 2 provides an overview of the fundamentals in distributed processing systems, focusing on operator placement and migration in a distributed CEP system, as well as the concepts of human relationships and trust in distributed computing. We classify the state of the art in these categories and detail the key research gaps, which are the focus of our work. Based on this discussion, we present the design of the proposed system as well as key design considerations in Chapter 3. Chapters 4, 5, and 6 present and describe the three main contributions of this thesis. Chapter 4 focuses on the estimation of the type and strength of human relationships based on past communication behavior using our smartphone application, FAMAPP. This forms the basis for the trust-based approach TRUSTCEP for distributed CEP, which is detailed in Chapter 5. Chapter 6 delves into FLEXCEP, which accounts for reliable CEP through flexible operator migration in dynamic D2D environments. Each of these chapters also entails an exhaustive evaluation of the respective approaches. Finally, Chapter 7 concludes this thesis with a summary and outlook towards possible future work.

2

BACKGROUND AND STATE OF THE ART

Our work focuses on the efficient placement and execution of CEP operators in a distributed environment, where we comply with the privacy constraints of the users involved, and take the dynamic characteristics of the environment into consideration. In the following, we present relevant information on the fundamentals behind our system as well as the state of the art on existing mechanisms towards privacy-aware distributed CEP in dynamic user environments.

Going by the three research questions introduced in Section 1.2, we divide this chapter into three main sections to address the different aspects on event processing (Section 2.1), trusted computing (Section 2.2), and human relationship analysis (Section 2.3). In turn, we focus on the following questions with respect to the state of the art: (i) How do existing mechanisms towards operator placement and migration in distributed event processing systems address privacy and reliability? (Sections 2.1.1, 2.1.2, and 2.1.3); (ii) How is the concept of (user) trust used to facilitate privacy-awareness in (distributed) computing? (Section 2.2.1); and (iii) What are the existing approaches to infer the relationship between two users and what are their main revelations? (Sections 2.3.1 and 2.3.2). Each of these sections is concluded by a short discussion of the identified research gaps, which form the basis for our work in the following chapters.

2.1 EVENT PROCESSING SYSTEMS

In recent years, there has been a steady rise in the number of applications that continuously process data originating from distributed sources and *on the fly* to produce necessary higher-level information about the data. Examples of such applications include, among others, environmental sensing applications that monitor the raw atomic data produced by sensors in the environment and produce higher-level situational information for context-aware services (as in the scenarios described in Section 3.1), financial stock markets that provide real-time analytics on the stock prices, and intrusion detection/security monitoring that help in detecting abnormal system behavior and thus identify potential attacks [28, 36, 45, 46, 53, 136].

In general, such applications are considered part of the so-called *Information Flow Processing* (IFP) systems [53]. In particular, CEP systems or just *event processing systems* are a class of systems that observe data flow in the form of a continuous stream of events. The focus here lies in the discovery of inherent patterns in multiple incoming events, and the production of requisite higher-level (complex) events to interested parties. In the following, we provide a short overview of event processing systems, compare them against other event-based systems like publish/subscribe and data stream management systems, and analyze their application in D2D-based environ-

ments. Subsequently, we proceed to analyze the state of the art in distributed event processing systems, especially targeting approaches for the placement and execution of processing operations on available devices.

Introduction to Event Processing Systems

Before we proceed to understand the functionality of event processing systems, let us first define what an *event* is, adopted from the definition stated by Etzion and Niblett [69] as well as Chandy et al. [45].

> EVENT. An *event* is an occurrence of any meaningful change that has happened or surmised to have happened within a particular system or domain.

In general, both the occurrence as well as a notification of the occurrence are termed *events* in related literature. The above definition limits the unlimited number of occurrences within a given system to those that are *meaningful*, depending on the application [36]. In general, an event can be two types: simple and complex.

A *simple event* can be a *change event*—where the value of a certain observed parameter differs from that in the previous observation, e.g., the position of an object—or a *status event*—where we observe the value of a certain parameter at a later point in time, e.g., readings of a temperature sensor. In the case of the latter, even same parameter values at different time instants can be considered a *meaningful change* depending on the application at hand, e.g., the internal temperature readings of a CPU.

A *complex event* is obtained by either a composition or a derivation based on multiple simple and/or other complex events. They are produced through operations of event algebra, e.g., aggregation, sequencing, or by using semantic knowledge on the occurrence of multiple simple/complex events. The aggregate sum of temperature readings of a CPU over a time period of 10 minutes is an example of a complex event based on composition. On the other hand, the inference that the CPU fan is not working properly, based on observations of increasing temperature readings over the same time period, is an example of a complex event based on derivation. In general, complex events are generated based on *Event-Condition-Action* (ECA) rules, which are applied on the incoming simple/complex events [36, 53].

A CEP system consists of three main components—a set of event producers, a set of event consumers, and an event processing engine. Figure 1 shows a component-based overview of a (distributed) CEP system. *Event producers* are the information sources in a CEP system, such that they monitor the environment pertinent to the application at hand (e.g., network traffic for intrusion detection) and produce the events based on their observations. *Event consumers* are the stakeholders of a CEP system, i.e., the ones interested in obtaining higher-level events produced based on the discovered patterns in (atomic) events generated by event producers.

The *event processing engine* forms the central component of a CEP system and is responsible for the transfer of the relevant events after appropriate transformation to the interested consumers, based on ECA rules specified by the application at

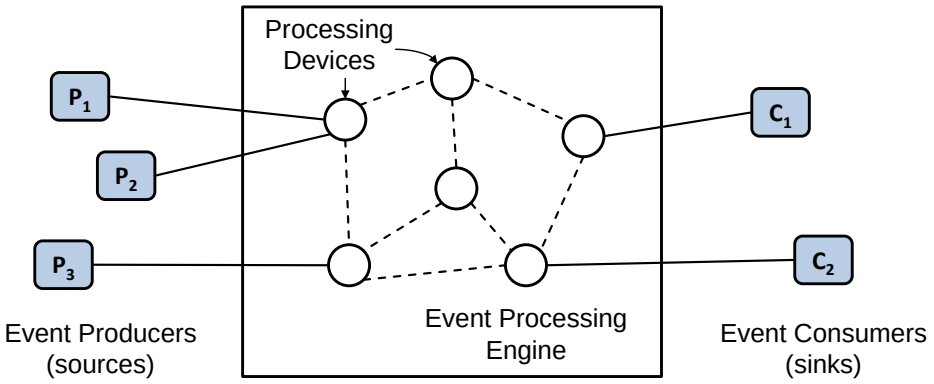


Figure 1: Main Components of a (Distributed) CEP System
(adopted from Cugola and Margara [53])

hand. The ECA rules are generally specified in the form of so-called *operators* that are the functional modules of a CEP system. Each operator accepts event streams as input, processes them based on the specified logic to detect higher-level events, and produces an output event stream that forms the input for subsequent operators. The event flow in the event processing engine is defined by connecting one operator to the next in the form of a graph called the *operator graph*.

The architecture of the CEP engine can vary in accordance with the application design. The event processing engine can be executed on a central entity (centralized) or on a distributed set of processing devices, commonly called *brokers* [39, 40, 70] or *event processing agents* [69]. In our work, we primarily focus on distributed CEP systems that are composed of multiple event processing devices in the CEP engine between producers and consumers. Each of these systems vary in their mechanisms for routing and forwarding events between the processing devices [41, 69], and how the processing functions, i.e., the operators, are distributed among the different devices [9, 167].

Parallels to Other Event-Based Systems

CEP systems have a lot in common with other event-based systems that are also based on the IFP principle. Two of the prominent examples are *publish/subscribe systems* [70] and *Data Stream Management Systems* (DSMS) [15] or *Stream Processing Systems* (SPS) [14], which are also based on the principle of message-oriented interaction. All three systems focus on the scalable handling of incoming data streams and events from heterogeneous sources, in accordance with a (large) number of processing rules to produce higher-level information, as queried by interested stakeholders.

DSMSs are based on the concepts behind active database systems, which comprise a persistent storage unit that maintains all relevant data. Similar to CEP systems, a DSMS applies ECA-based application rule logic in the form of queries on stored data in either a one-time or a continuous basis to produce the required higher-level

information, which is again stored in the database for further processing [15]. These queries generally entail simple filtering operations or join operations over two or more incoming data streams. However, unlike CEP systems, DSMSs do not focus on the discovery of hidden patterns in the incoming data streams through appropriate sequencing and ordering [53].

On the other hand, the publish/subscribe paradigm provides a generic protocol design for the realization of event-based systems with the help of a *Message-Oriented Middleware* (MOM). Similar to CEP systems, publish/subscribe systems consist of *event publishers*, *event subscribers*, and an *event notification service* [70]. In publish/subscribe systems, the publishers and subscribers are decoupled in space and time, such that they do not need to know each other and need not be online at the same time [40, 70, 165]. Subscribers express their interests in the form of subscriptions with respect to the events generated by the publishers. The event notification service—which can be a set of brokers, just as in the case of CEP systems—primarily filters the incoming published events on the basis of the existing subscriptions, and sends the relevant events to the respective subscribers. Similar to CEP systems, there can be complex event filters necessary to satisfy the prevalent subscriptions based on the content of the published events [166]. In this respect, CEP systems can be perceived as an extension to publish/subscribe systems with additional complex operations to detect further hidden patterns in the (raw) events produced by the information sources [53].

CEP over D2D

Moving towards 5G cellular networks (and beyond), one of the paramount technologies expected to have a strong impact on the scalability, latency, and bandwidth utilization in cellular networks is *Device-to-Device* (D2D) communication. Cellular networks with this functionality can allow two devices in close proximity to communicate directly with each other and process information cooperatively, in a self-organizing manner, without the involvement (or with minimal involvement) of cellular base stations [31, 140, 211] (see Figure 2). D2D communication can play a pivotal role in the context of modern communication advancements, such as fog computing [34, 35, 94, 157, 214] and cloudlets [185, 215], as well as MANETs [3] and *Vehicular Ad-hoc NETworks* (VANETs) [90]. Its application use cases extend from typical network off-loading in big malls or stadiums to areas of natural disaster without possibly any cellular connection.

In the IoT, D2D communication forms the basis for collaboration among different Internet-compatible devices in the environment, facilitating the deployment of CEP systems on top to allow for the detection of higher-level situational and contextual information. Devices communicate with each other using short-range networks based on underlying communication protocols like WiFi, Bluetooth, Zigbee, and RFID, thus forming multiple-hop networks [31, 140]. In turn, CEP over D2D allows for *in-network processing* of atomic events from the producers, without the involvement of centralized processing entities, especially in view of the resource-constrained and mobility-driven nature of devices in the IoT. Initial research work

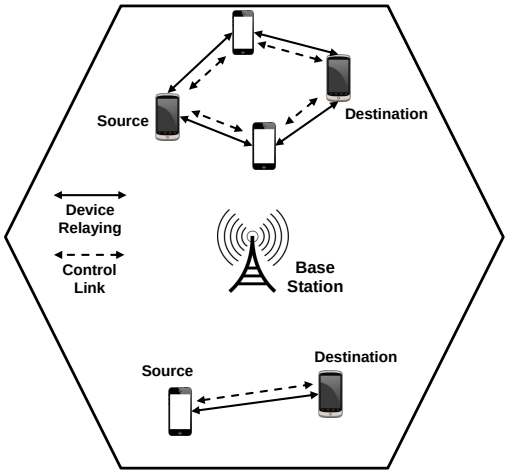


Figure 2: A Network based on D2D Communication
(adopted from Tehrani et al. [211])

on the use of CEP in wireless sensor networks proves its viability as a processing middleware in such resource-constrained environments [122, 183].

Recall from Chapter 1 that one of the central design considerations for any distributed CEP system is the placement of CEP operators on appropriate devices. This becomes increasingly important in D2D-based networks in the IoT, where the devices have varying mobility patterns and resource availability, such that merely placing the operators does not guarantee a reliable processing environment. Above all, given the sensitive nature of the data exchanged in an IoT scenario, it behooves the distributed CEP system to comply with the (possibly varying) privacy constraints of the users involved. In view of these considerations, we now analyze the state of the art in distributed CEP systems to understand how they account for the above-mentioned challenges.

2.1.1 *Operator Placement in Distributed Event Processing Systems*

Most approaches towards distributed event processing focus on optimizing operator execution by placing the operators on the appropriate devices, such that the application-specified criteria, e.g., latency considerations, are satisfied. The most noticeable criteria for operator placement in existing event/stream processing systems are latency, bandwidth considerations and network load [38, 121].

Rizou et al. [176] propose a placement algorithm that accounts for the reduction of end-to-end latency in large-scale stream processing environments, while also keeping the network load at a minimum. They give a higher preference to nodes with more *residual* resources for the placement of operators, if they lead to the reduction of the overall processing delay. Similarly, Pietzuch et al. [167] devise a *Stream-Based Overlay Network* (SBON) to determine a decentralized strategy for operator placement. By maintaining a distributed *cost space* metric, SBON determines the place-

ment strategy that induces the least network usage, while maintaining a low latency and improved bandwidth utilization. Hong et al. [95] and Ottenwälder et al. [157] propose approaches to provide for low-latency processing of CEP queries in *Mobile Situation Awareness* (MSA) applications. They propose algorithms to cover moving range queries and spatio-temporal event processing that uses a prediction-based continuous query handling.

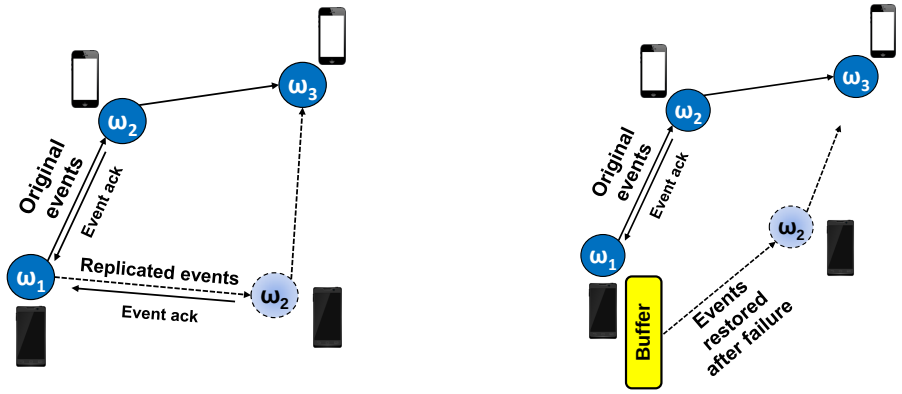
Towards network load balancing, Cugola and Margara [54] propose different deployment strategies for distributed CEP, based on their own CEP query language called TESLA. Their approach allows for reduction of processing load across a multiple processor architecture, while maintaining the correct operation of the distributed CEP system. Jayasekara et al. [108] propose a load-balancer based CEP system called WIHIDUM that allows for the distribution of CEP operators among the available nodes, by breaking the CEP query into multiple sub-queries. They propose multiple algorithms including pipelining and partitioning towards scalability of distributed CEP systems. Mayer et al. [134] propose an improved windowing approach for parallel CEP, such that they minimize the processing load as well as the communication overhead when placing the operators.

Energy-efficiency also plays an important role in operator placement, especially in resource-constrained D2D-based networks like *Wireless Sensor Networks* (WSNs) and MANETs. Focusing on running a CEP system on a WSN, Li et al. [122] attempt to reduce the energy consumption on the individual nodes by splitting the complex queries into server-side and node-side queries, such that only real-time events are processed directly on the nodes. By doing so, they claim to improve the real-time performance of their system. Yang et al. [231] propose a collaborative query processing framework to utilize the processing power of the individual devices, and minimizing the amount of communication with a centralized server. They show that their approach allows for a reduction of 60% of the overall energy consumption overhead in the system. Starks et al. [201] aim to minimize energy consumption in resource-constrained devices by limiting the data transmission costs and achieving near-optimal operator placement, compared to a centralized approach.

2.1.2 *Reliability in (Distributed) Event Processing Systems*

All of the above approaches deal with the initial placement of CEP operators, but do not address the problem of operator graph maintenance upon changes in the system or the network. Operator migration is a key requirement in systems where the network undergoes dynamic changes, e.g., due to node[1] failure or mobility, such that the operator graph can be restored and system reliability can be maintained. Most approaches towards reliability and fault-tolerance in event processing systems can be put into two main categories—(active) replication mechanisms [22, 84, 86, 157, 188, 217] and rollback-recovery mechanisms [42, 65, 85, 100, 101, 116]. Replication mechanisms maintain one or more backup nodes at all times and improve system reliability by masking node failures. Rollback-recovery mechanisms, on the other

1 In the context of reliability, we switch between the terms *node* and *device* interchangeably.



(a) Illustration of Active Replication

(b) Illustration of Rollback Recovery

Figure 3: Illustration of the Two Traditional Reliability Approaches—Active Replication and Rollback Recovery/Upstream Backup

hand, use a scheme of persistent event storage on the nodes at different checkpoints such that these can be used for restoring the system upon failure. Figure 3 illustrates these two traditional approaches.

Active Replication

With respect to fault-tolerance systems, Schneider introduced active replication mechanisms to combat Byzantine and fail-stop failure models [188]. Guerraoui and Schiper [86] furthered this concept towards software-based replication, where they emphasize on the necessity for group communication and consensus algorithms to facilitate software-based replication on off-the-shelf hardware. In general, such systems introduce a *horizontal* degree of redundancy in the operator graph execution by replicating the operators on multiple nodes (see Figure 3a). In turn, they minimize the delay penalty (loss of events) and recovery time during migration, since the redundant paths can be chosen upon failure of the main path of the operator graph. However, this introduces a high amount of communication overhead and processor utilization, since all event streams must be replicated on the backup nodes.

Balazinska et al. [22] propose an approach called DPC (Delay, Process, and Correct) to handle failures on the part of processing nodes and network failures in the Borealis stream processing system [1]. Using DPC, the authors attempt to minimize the inconsistencies across all replicas, by allowing for an appropriate trade-off between data availability and consistency. Heinze et al. [92] propose an algorithm to optimally adapt a stream processing system through appropriate scaling in and scaling out, depending on the dynamic nature of the workload. They evaluate the principle of auto-scaling a stream processing system by evaluating different strategies to determine the appropriate threshold for adaptation.

Towards active replication in CEP systems, Völz et al. [217] propose an algorithm that tolerates n possible failures by introducing n redundant replicas for each oper-

ator. Ottenwälder et al. [159] focus on *Mobile CEP* (MCEP) in the context of MSA applications, and propose a probabilistic migration approach. Their main focus lies in reducing latency and improving bandwidth utilization in mobile scenarios through appropriate migration of CEP operators between edge computing nodes.

Other approaches focus on managing the run-time functionality of operators or in selective replication to minimize communication costs. Mutschler and Philippsen [145] propose a framework for runtime migration towards reallocation in distributed event-based systems. They propose a cooperative handover mechanism to identify the host devices that receive the replicated event stream from predecessor devices. Grüneberger et al. [84] propose a conceptual idea towards a selective replication approach in CEP systems, where they replicate only selected fine-grained parts of an operator. In turn, they claim to improve system availability with minimal resource usage. Ottenwälder et al. [158] propose RECEP, an approach to reuse partially overlapping operators in mobile CEP systems. In turn, they attempt to minimize the computational overhead and resource consumption.

Rollback Recovery/Upstream Backup

Rollback recovery or upstream backup is an approach for restoring previous checkpoints or logs during the execution of any processing system, as summarized in the survey by Elnozahy et al. [65]. This concept has also been proposed in the context of network middleboxes by Sherry et al. [191]. They propose improved algorithms to manage the state of middleboxes (e.g., activity logs and port mappings) upon potential network failures, by reducing the amount of logged information per middlebox.

In the context of event processing, rollback recovery is mainly done by storing past events on the dedicated buffers on the devices and retrieving these events upon failure, therefore introducing a degree of *vertical* dependency in the operator graph execution (see Figure 3b). Generally, this is used in large centralized event (or stream) processing systems, where the storage of events can be controlled in a flexible and controlled manner. This concept was first introduced by Hwang et al. [100] where they use upstream nodes as backups for their downstream neighbors by logging event tuples in their output buffer. Similarly, Gu et al. [85] focus on improving checkpointing by reducing the communication overhead during run-time through the usage of cumulative acknowledgments.

Certain other approaches focus on the state of operators in the processing chain to optimize system recovery. Castro Fernandez et al. [42] introduce a scale-out and fault-tolerance mechanism for stream processing in cloud computing. They propose the externalization of internal operator state to optimize the usage of *Virtual Machines* (VMs) based on varying system load, and do so by exploiting the processing, buffer, and routing states of each operator. Koldehofe et al. [116] approach the same issue in event processing systems by developing an adaptive acknowledgment-based mechanism to maintain non-reproduceable operator state in the form of *savepoints*. They incorporate the operator state in their approach to identify the points in time where the amount of state information is minimal.

Building on their previous approach, Hwang et al. [101] propose a self-configuring, cooperative approach for delta-checkpointing with stateful operators like `AGGREGATE` and `JOIN` in distributed stream processing systems. They show that their approach performs better than approaches that employ active replication or complete checkpointing in terms of recovery time and run-time overhead. Wang and Peh [219] develop MobiStreams, that deploys a distributed SPS on smartphones. They tackle the problem of multiple failures and network instability by employing improved checkpointing techniques—a token-based approach that allows for each operator to coordinate its checkpointing and thus reduce the amount of stored data; and a broadcast-based approach that uses UDP-based data transmission to reduce unnecessary network overhead.

Hybrid Approaches

Some authors have proposed hybrid approaches by combining different fault-tolerance mechanisms, primarily in the context of cloud-based stream processing systems. Zaharia et al. [236] propose a stream processing model called *discretized streams* to overcome both system faults as well as slow-processing nodes or *stragglers*. Heinze et al. [93] propose an adaptive approach for active replication by scaling the system in and out, elastically. By using recovery time estimates for each operator, they determine the required number of operators for replication. Su and Zhou [204] propose fault-tolerance mechanisms for massively parallel stream processing systems. They use active replication for only certain operators depending on the availability of resources, and propose a modified passive replication approach for the others. By generating *tentative* outputs for the operators without active replication, they attempt to find the optimal balance between active and passive replication schemes.

2.1.3 *Privacy in Event-Based Systems*

The issue of privacy gains paramount importance in the field of the IoT, considering that the data generated by the different sensors can be extremely sensitive. Sensor data can reveal private information about users, such as their location or activity, which in turn can be used by malicious users to extract higher-level information about the users, such as their apartment occupancy hours or their stress levels [184, 193].

Combating privacy becomes increasingly difficult in event-based systems, given the fact that the data generated by producers can be delivered to any consumers based on the processing rule logic of the event processing/notification service. This sense of space decoupling, which is especially inherent to publish/subscribe systems, leads to many complications when dealing with user privacy. Are the consumers trustworthy? Is the processing/broker system trustworthy? How can the higher-level events be observed without violating the privacy constraints of the producers?

In this section, we analyze the different methods used in related literature on event-based systems—and primarily, CEP systems, DSMSs/SPSs, and publish/subscribe systems—to combat privacy and security attacks, in general. We focus on a selected

set of approaches that deal with the confidentiality of information exchanged and the protection of user privacy in distributed event-based systems. For a complete survey on confidentiality and security in publish/subscribe systems, we refer the interested reader to the work by Esposito and Ciampi [68], and Onica et al. [155].

Encryption-Based Mechanisms

Overall, *Public-Key Infrastructures* (PKIs) are commonly used on a wide scale for security purposes in any distributed system, where a trust certificate authority facilitates encrypted communication with the help of public and private keys [6]. This also forms the basis for the popular encryption technique, *Transport Layer Security* (TLS), that is used for privacy-preserving event dissemination in a few event-based systems [18, 194, 197]. However, these approaches assume the processing nodes in between to be trusted.

Many encryption-based mechanisms in distributed event-based systems focus on the confidentiality of the data transmitted across the broker/processing network, such that the processing agents/brokers cannot read into the contents of the data exchanged in the system. In most approaches, *honest-but-curious* processing devices are considered, such that they follow system protocol in an honest manner, but are also curious to learn as much as possible during protocol execution.

Shikfa et al. [192] analyze the different confidentiality requirements in *Content-Based Publish/Subscribe* (CBPS) systems, and present a secure routing algorithm to disseminate events efficiently. They propose encrypted routing tables based on multiple layer commutative encryption, and secure look-up operations to allow brokers to perform their forwarding operations on encrypted data. Barazzutti et al. [26] introduce a pre-filtering mechanism in publish/subscribe systems, which builds on privacy-preserving encrypted matching schemes. In doing so, they propose a computationally less intensive approach that leverages the existing relationships between the subscriptions to minimize the encryption overhead and yet support data confidentiality. Tariq et al. [210] propose authentication and confidentiality in broker-less CBPS with the help of identity-based encryption. Using pairing-based cryptography mechanisms and fine-grained key management, they enable subscription confidentiality and efficient routing of encrypted events.

De Oliveira et al. [152] propose an approach for privacy-preserving event correlation using symmetric homomorphic encryption techniques. They particularly focus on resolving *greater-than* and *range* queries in the context of stream processing events, such that the processing devices can perform query resolution but do not infer stream contents. Nabeel et al. [147] propose a novel cryptographic approach—based on the Paillier homomorphic cryptosystem [162]—to protect both data confidentiality and to safeguard subscriber privacy in CBPS systems. They mainly focus on performing the operations `MATCH` and `COVER` over the brokers, without revealing the contents of the subscriptions and notifications to them.

In their follow-up work, Nabeel et al. [146] extend their previous work by a novel approach for group key management in *context*-based publish/subscribe systems. This allows for fine-grained encryption-based access control, so that the context-

based privacy constraints of the publishers and subscribers can be supported. Instead of *Trusted Third Party* (TTP) servers performing subscription encryption like in their previous work, here, the authors employ so-called *context managers* that provide the required security parameters to the publishers and subscribers—based on their current context—to encrypt their notifications/subscriptions. Based on the group key management system, the context managers control the information level exchanged among the subscribers.

Access Control Mechanisms

Access control mechanisms are normally used as an added security mechanism on top of encryption schemes, like the ones described above. In general, any encryption scheme involves the exchange of keys to decrypt the encrypted data and retrieve the original message. Access control mechanisms account for user authentication and authorization, so that the keys can be exchanged according to the amount of data a user is allowed to read [68, 155]. User authentication entails the verification of a user's identity, whereas authorization accounts for the rights given to a user to access certain data or to use a certain functionality in a system. Access control mechanisms have been proposed in the field of cloud computing [235], online social networks [48], where the general notion is to allow access to user data based on the governing users' policies.

Belokosztolszki et al. [32] propose a *Role-Based Access Control* (RBAC) mechanism for publish/subscribe systems, which allows users to manage their data by grouping the other users into different roles based on how trusted they are. They use the event-based middleware called Hermes [165] to create a secure framework called Oasis, which implements policies for: broker-client interaction, type management (i.e., which roles are allowed to access event types), as well as advertisement and subscription rights. In this system, unlike the above approaches with homomorphic encryption schemes, only certain brokers are assumed to be untrustworthy, so that certain trustworthy brokers can manage the role-based certificate allocation.

Another popular access control mechanism is *Attribute-Based Access Control* (ABAC), where the authorization to read and modify data is assigned to users based on certain data attributes as well as attributes of the users themselves [98, 99]. This allows for a more fine-grained and flexible mechanism for access control, such that additional factors like environmental conditions or user context can be taken into consideration while allocating authorization rights. Fiege et al. [73] introduce the concepts of ABAC in the publish/subscribe system Rebeca [163], where they manage the *scope* of publisher notifications with the help of attribute certificates. In this approach, a distributed infrastructure manages the attribute-based keys using an overlay network and is assumed to be trusted by all users.

In the field of event and stream processing systems, Lindner and Meier [126] propose a secure stream processing framework that employs access control to manage the data flow. They implement their approach on top of the Borealis engine, developed by Abadi et al. [1], which focuses on query processing and optimization in centralized DSMSs. Their access control mechanism accounts for the authorization of

users to determine who may receive sensitive data streams. To this end, they employ a simple tuple-based stream suppression method that allows for privacy-preserving processing of data streams.

Information Flow Control

Another approach towards security in distributed processing systems is a data-centric approach called *Information Flow Control* (IFC), that implements *mandatory* access control. Basically, IFC deals with the tracking of data (streams) in a system, such that the propagation of data is controlled through appropriate security labels that are applicable throughout the entire system, without allowing the users to modify them (unless they have the rights to do so) [19, 139, 194, 195]. IFC focuses on data isolation, data flow tracking, and data flow enforcement between any two users in a given system. In other words, while normal *discretionary* access control mechanisms—like the ones discussed above—consider user-specific policies, systems using IFC consider a system-wide administrative control [19].

The initial work towards IFC in publish/subscribe systems was developed by Singh et al. towards policy-based information sharing [197] and IFC in healthcare environments [196]. Building on these initial approaches, they developed an approach called interaction control (IC) for publish/subscribe systems based on legal obligations in a given application environment [194]. Their approach incorporates granular security constraints that allow for inter-domain event dissemination in a privacy-preserving manner, without allowing non-domain brokers to obtain any sensitive data from within a certain domain.

Migliavacca et al. [139] propose a lightweight IFC framework for event processing systems in the form of DEFCon. They enforce information security by introducing security labels or *tags* to the event messages (or parts of the event messages)—a *confidentiality* tag dictates where an event (or event part) can flow to; an *integrity* tag accounts for where an event can flow from. Using these security tags and additional privileges that can be assigned to certain users, the DEFCon approach accounts for low processing latency and yet improved security than traditional approaches. However, the main drawback of this approach is that it is based on centralized systems where a single administration has control over the individual system components. The security labels are controlled and assigned in isolated domains that are defined at the OS-level within these systems, and cannot be modified otherwise.

Enck et al. [67] address the issue of information disclosure within current smartphones among third-party applications, and propose a system called TaintDroid that provides an efficient and dynamic service for tracking sensitive data on smartphones. TaintDroid labels the sensitive data from private sources—e.g., contextual information—and tracks their propagation through program variables, files, and inter-process messages through the *taint* they leave behind. Thus, TaintDroid allows users to obtain real-time information about the data that the different mobile applications use at any point in time. In turn, their approach allows for applications to share information under the privacy constraints established in the system. Their

evaluation results show an increase of 32% with respect to the processing overhead on a CPU-bound micro-benchmark.

Obfuscation Mechanisms

Some other related work focuses on the obfuscation of certain data events when executing CEP operators in a distributed manner on different devices, such that the sensitive input events cannot be inferred by observing the obtained output events. This goes hand in hand with the above mechanisms where the obfuscation of the disseminated data events can be done in accordance with the access control policies and information flow control.

The existing approaches towards *Privacy-Preserving CEP* (PP-CEP) predominantly lay focus on static operator graphs and/or scenarios with a low degree of dynamics. He et al. [91] present a concept for PP-CEP by analyzing the implications of event pattern reporting on user privacy. By grouping the generated event patterns into public and private patterns—those that should and should not be reported, respectively—they propose a mathematical foundation for PP-CEP such that private patterns can be suppressed intelligently without compromising on the utility of the CEP system. They primarily concentrate on facilitating privacy-preserving event reporting in the case of SEQUENCE and WINDOW operators, such that the adversaries may not be able to infer the private events based on the externally observable events.

Schilling et al. [187] approach this issue by considering user access policies that are used to obfuscate private event patterns over a chain of dependent operators in a distributed CEP system. They mainly focus on developing a scalable method to measure the obfuscation imposed by access policies, especially when there are multiply-connected correlation paths. Their work is based on the analysis of `AGGREGATE` operators, such that the input sensitive events can be sufficiently obfuscated by the output observable events.

2.1.4 *Discussion*

Recall that, in our approach, we focus on preserving the privacy of the disseminated events in a distributed mobile CEP system, where the participating users and their devices—i.e., their resource availability and the privacy constraints of the users—can vary dynamically with time.

In terms of reliability, none of the above approaches deal with dynamic D2D environments where the nodes are mobile and resource-constrained. In our initial work on CEP over D2D [60], we explored the possibility of migrating CEP operators in D2D-based networks by analyzing the main challenges. However, the proposed approach in that work does not account for mobility-awareness and appropriate buffer management. In our approach, we propose a flexible fine-grained operator migration approach that employs the above-described fault-tolerance and reliability mechanisms in accordance with the network conditions. In doing so, we leverage the advantages of active replication and rollback recovery to reduce recovery time as well as run-time communication overhead in distributed CEP systems.

In terms of privacy, most of the above approaches are not feasible in dynamic scenarios, since these scenarios do not provide *a priori* knowledge of the producers, consumers, and intermediate processing devices in the environment. The usage of homomorphic encryption—while promising for privacy-preserving operations in a distributed environment—has not yet been proven to be applicable for wide-scale deployment, especially in the IoT [97, 195]. To account for these dynamic conditions, we propose an approach that incorporates the inherent trust among the participating users, such that collaborative processing can be facilitated without violating user privacy constraints. This allows the other approaches towards access control and obfuscation mechanisms to be used in parallel with ours, where we identify the trustworthiness of the collaborating devices to establish the user access policies.

2.2 TRUST IN COMPUTING

A major component of human relationships and any social network is trust. The most common form of trust in research is interpersonal trust, where trust is seen as a medium to evaluate the strength of a social tie between a *trustor* and a *trustee* [133]. In general, trust between two people—*dyadic trust*—depends on their interactions over time, and in turn, also influences the behavior of both people in a relationship [113, 181].

In the field of computer science, trust is normally differentiated into *system* trust and *user* trust. System trust mainly describes the trust or expectations placed on a computer system or device, such that it fulfills the purpose it is intended for [190, 232]. In our work, we account for the aspect of system trust by proposing reliability mechanisms, as mentioned above in Section 2.1.4.

Our primary focus lies in the evaluation and management of *user* trust. User trust finds its roots in the fields of psychology and sociology. One of the most widespread definitions of user trust, as defined by Gambetta [75] and transcribed by Quercia et al. [168], reads, "[trust] is a particular level of the subjective probability with which an agent will perform a particular action, both before [we] can monitor such an action and in a context in which it affects [our] own action." Basically, user trust describes the level of confidence a person places in another person for a particular action, without them having the ability to control the execution of the action itself [168, 190].

In general, trust[2] *evolves* with experience within a relationship. A positive experience strengthens the trust in a relationship, whereas a negative experience depletes the same. Trust in human relationships satisfies a set of properties, which must be dealt with when incorporating trust in computing systems.

Subjectiveness. People exhibit distinct behaviors and perceive things around them differently to one another. This leads to the fact that one may have different thresholds for their trust levels towards others. Wang and Redmiles [221] define this inherent tendency to perceive the trustworthiness of other people as the *baseline trust* of

2 In the following, unless otherwise mentioned, we simply use the term *trust* to refer to user trust.

an individual. Trust is, therefore, predominantly influenced by one's predispositions towards others and the services offered by them.

Propagation of trust. Trust between people is based on its propagative nature [190]. The trust placed by person A in person B also depends on the trust placed by person C in person B, as well as the trust placed by person A in person C. While trust is not transitive [190], in that there does not *have* to exist any trust between A and B, there can exist a propagation of trust due to the social network between A, B, and C.

Context-dependence. A person's location and surroundings can have a significant effect on their trust towards others. People generally trust their co-workers only within the confines of their work environment, whereas they may not trust their family members when it comes to confidential work-related matters. This also plays an important role in establishing trust based on propagation.

Dynamism. Trust improves with increase in successful operations between two people, and decays over time and with negative experiences. Given that user behavior is generally very dynamic in nature, with varying interaction patterns, the *evolution* of trust is an important aspect for any trust-based approach.

Asymmetric. In principle, trust between people is not necessarily symmetric, i.e., the trust placed by person A in person B need not be the same as that placed in person A by person B. The asymmetric nature of trust between two people can reduce with time and positive experiences, leading to same trust levels in both directions. Similarly, any trust aberrations by person A will eventually lead to a decreased trust by person B in A.

A general model for trust management comprises the *evaluation* of the direct trust between two people, as well as its *evolution*. The direct trust between two people is their trust based on the direct interactions between them. Trust evaluation is the process of calculating this trust based on certain measurable trust metrics that quantify the quality of the services offered by one to another. To better understand these concepts, let us consider the following terminology based on the above definition for trust. Users A and B (the *trustors*) have distinct levels of trust towards user C (the *trustee*) with respect to a particular set of actions X in context Y.

Trust evaluation is the process of calculating trust on C based on certain trust metrics. These metrics are measurable parameters based on X and Y that quantify the quality of the services offered by C to A and B. For example, a file sharing application may have trust metrics defining the quality of the file received, based on the file size and its content. A wrong file size or empty contents offered by C will lead to a negative trust level from A or B. This assessment of trust based on own experiences in termed *direct trust*. On the other hand, A may evaluate its trust towards C based on B's recommendation. This is termed as *recommendation trust*. These two trust parameters allow for the calculation of the *overall trust* of A towards C. It is necessary to account for the same set of actions X and the context Y while evaluating the trust. Since user trust is asymmetric, the above procedure has to be performed independently for each user in the environment.

The robustness of one such trust-based approach requires an efficient mechanism to update trust values at regular intervals, accounting for *trust evolution* [168, 190].

Trust evolution accounts for the dynamism that characterizes user trust. Varying experiences as well as recommendations from others need to be considered while evaluating the trust level over time. This allows the trust framework to adapt itself to the existing changes in user behavior as well as interaction patterns. Trust recommendations allow for the adaptation of existing trust values, so that the trust values converge towards the optimal solution. However, one major problem here is the possibility of falsified and inaccurate recommendations. Malicious users can use this approach to modify the system performance to suit their needs. It is necessary to identify false recommendations and take appropriate measures to deal with such situations.

2.2.1 *Trust-Based Models for Distributed Systems*

Given the above introduction to the notion of trust in computing systems, we now present the state of the art of trust-based models in distributed systems. We particularly focus on existing mechanisms for the evaluation of trust in different distributed systems, including P2P and D2D systems, to analyze how they tackle the above-mentioned challenges. Furthermore, we also present how these trust-based systems overcome the presence of adversaries.

Trust Evaluation

The first and most important step in any trust-based system is the determination of the trust between communication parties, i.e., *trust evaluation*. Any trust management model attempts to establish the trust existing between a trustor and a trustee by analyzing certain metrics in an appropriate manner, depending on the application settings and the methods of interaction that can be evaluated. The application settings vary from pervasive computing environments and social networks to P2P and ad-hoc networks. Most trust management models also avoid the usage of a centralized management service, given the fact that one such service can be compromised as a single point of failure/attack.

Mui et al. [144] propose a computational model for trust and reputation based on a Bayesian formalization that accounts for context-dependent and subjective trust. In turn, they propose a model that allows for a probabilistic inference of trust and reputation, such that it can be implemented in a real-world multi-agent[3] environment. Liu and Xiu [127] also focus on multi-agent networks and propose a trust negotiation framework that allows client systems to establish the trust relationship with the system. This allows the system and clients to verify each other's credentials in an automated manner with the help of *security capsules*. The authors claim that their concepts can be applied to any distributed system and Internet applications.

Quercia et al. [168] build on the work by Mui et al. and put forth a computational trust framework for pervasive computing, incorporating the trust properties of sub-

3 In the context of trusted computing, the term *agents* is used to denote the stakeholding devices in the system.

jectiveness as well as time- and context-dependence. They propose a generic *n*-level discrete trust model, which is lightweight and allows for the incorporation of both direct and recommendation based trust metrics. Through extensive evaluations, they show that their trust model can be used to perform packet delivery in a network with possibly malicious peers.

Ries [175] proposes a decentralized computational trust model, allowing agents to interact with other partners based on their trustworthiness. He incorporates two representation schemes in his trust model—(i) a human trust interface (HTI) that accounts for the probability of trustworthy behavior of an agent in any future transaction and the context-based certainty of the said probability; and (ii) a Bayesian approach using probability density functions, similar to previous work by Mui et al. and Quercia et al. In doing so, he mathematically proves that a mapping between both the representation schemes is possible.

Moving to other applications, Jamali and Ester [106] propose a scheme for trust-based recommendation in recommender systems. Using a random-walk method, they propose an improved scheme called TrustWalker that recommends items to users not only based on the ratings of the items, but also based on the similarity among the different items. In turn, this allows new users in the network, who are not within an established trust network, to obtain appropriate items. Nepal at al. [149] focus on trust models in social networks by establishing trust communities based on a concept called *social trust*. Their social trust model STrust considers the social capital of a relationship, in that it accounts for the interactions, comments, and discussions between the users, to establish the trust.

Adali et al. [5] introduce the concept of *behavioral trust* in the field of social networks, which is a form of dyadic trust that is based on the communication characteristics between the two parties. They propose that the communication behavior between people—e.g., nature of phone conversations, e-mail correspondence, social network comments—can be used to quantify the trust between them. Subsequently, they propose models for conversational and propagation trust, and validate them on Twitter[4] datasets. Similarly, Trapp et al. [212] propose a sociologically-inspired model for trust in MANETs, which uses the social relationship between the mobile users to establish trust. They argue that reliable trust measurement based on network data requires a densely populated MANET. Instead, they propose the usage of the tie strength between users to establish trust, discussing the possibilities of assessing trust based on local smartphone interaction data.

Trust in P2P Systems

P2P systems are a branch of distributed systems that are, by and large, devoid of any hierarchy or centralized organization [202]. They are based on self-organizing overlay networks, where peers may act as both clients and servers, to form so-called *servents*. In doing so, peers provide access to their resources, such as documents, movies, etc., and support the sharing of these resources with the other peers in the

4 https://twitter.com

system. To this end, P2P systems are characterized by specialized routing algorithms, storage mechanisms, and scalability, among others, so that a cooperative framework of peers can be established and maintained [112, 118, 202].

One of the most important requirements of any P2P system is an appropriate mechanism for the evaluation of trust and reputation among the peers, so as to facilitate a secure and trustworthy communication system. Given the lack of a centralized organization, malicious peers can attempt to provide falsified data to incoming requests, impersonate other peers, perform *Denial-of-Service* (DoS) attacks, etc. Considering that peers in a P2P system engage themselves in bilateral communication, it is possible to develop models to evaluate computational trust and peer reputation through appropriate feedback mechanisms, which in turn allows benign peers to defend themselves against adversaries.

One of the first approaches to deal with trust and reputation in P2P systems was introduced by Aberer and Despotovic [2]. Building on general reputation-based models, they develop a scalable trust model that works without the need for a central organization and calculates reputation of peers based on their previous interactions with other peers. Peers can make and store their complaints about the services offered by other peers in an overlay network called the P-Grid. The P-Grid can then be accessed to estimate the trustworthiness of other peers. To improve scalability, replicas of complaints are stored on multiple peers, also allowing for robustness against failures in the system.

Kamvar et al. [111] present an algorithm called EigenTrust to manage the reputation of peers in a P2P network and thus, reduce the transfer and exchange of inauthentic files in the network. EigenTrust allows all peers to engage in calculating a global trust value of the other peers in a distributed and node-symmetric manner. To do so, with respect to a certain peer X, the local trust value of each peer towards X are combined with the trust values placed by the other peers towards X, using a weighted aggregation mechanism. The *left principal eigenvector* of the resulting matrix provides the global trust value of peer X. A peer uses these global trust values to decide whom to contact for any file in the P2P system.

Xiong and Liu [229] propose a reputation-based trust algorithm for P2P-based e-commerce systems called PeerTrust, where they likewise develop a transaction-based feedback system to estimate the reputation and trustworthiness of the peers in the system. For their algorithm, they include peer feedback, number of transactions, feedback credibility as the main trust parameters. Furthermore, they also include context-based parameters based on the transaction and the community at hand. Building on this model, Srivatsa et al. [200] develop TrustGuard, which introduces a generic framework for calculating a dependable trust model under dynamic changes in peer behavior. Their model accommodates for behavior fluctuations on the part of other peers, such that positive interactions lead to gradual trust increase, while negative interactions lead to harder punishments.

In his PhD thesis, Liebau presents a token-based accounting scheme to facilitate a trust-based communication and collaboration environment [124]. Tokens are used by peers as objects for permission and receipts, such that a token spent is equivalent

to a transaction in the system. Using an appropriate structure for the tokens as well as token aggregation mechanisms, the thesis shows how the trustworthiness of peers can be determined and used by other peers to ensure trusted computing.

Vidyalakshmi et al. [216] present a decentralized trust model to facilitate transfer of data in a (mobile) P2P network based on the trustworthiness of other peers as well as the given situational context. They introduce an access control mechanism using their trust model that controls access to files based on their category (music files, family photos, etc.) and context (location, time, light, etc.). Similar to the previous efforts described above, trust between peers is calculated based on historical transactions with regard to file transfers in a given context. In addition, recommendation-based trust is incorporated to improve trust values in case of lack of direct interactions.

Socially-Aware D2D Networks

The aspect of trust also plays a very important role in D2D-based networks, where devices discover and interact with each other without any (or with minimum) involvement of a centralized entity. To this end, a class of networks have been developed that incorporate the social structure among users in establishing a trusted environment for device-level interactions. Such networks are called *social-aware* (sometimes, *socially-aware*) D2D networks, where social behaviors of the users are leveraged to assist D2D communication in a privacy-aware manner [123].

Ometov et al. [154] implement a proof-of-concept for incorporating social trust associations over proximity-aware D2D communications. In doing so, they account for the dynamic variations in the networks and focus on the establishment of secure groups among the changing D2D network participants. They assume that a central entity (base station) handles the discovery of devices and also assumes the role of a trusted certificate authority in the system. Performance evaluation results of their prototypical implementation show that the computational latency lies in the range of 60-120 ms, depending on the dynamic nature of the environment.

Wu et al. [228] focus on optimizing the usage of different content sharing modes in D2D environments, switching between base-station-to-device (B2D), D2D, and multi-device-to-device (MD2D) modes. To this end, they introduce a *social-aware rate* at which the devices should communicate in order to ensure high physical link quality and effective cooperation among the devices.

Militano et al. [141] deal with the coalescence of *Narrowband IoT* (NB-IoT) and D2D communication, moving towards 5G cellular systems. They present trust models to determine the reputation and reliability of devices in D2D-based content uploading networks, thus enabling *social awareness* among the devices. Similar to the work by Ometov et al., the eNodeB (a base station in LTE-Advanced cellular networks) stores the information about device reliability, reputation, and trust in the system. Each devices refers to this information to decide on the social reliability and recommendation reliability of the remaining devices.

Mitigating Adversary Attacks in Trust-Based Systems

Security mechanisms are developed in order to protect systems against adversary attacks. This becomes even more important in trust-based systems, given the intangible and probabilistic nature of trust between humans. Furthermore, as observed in the above approaches, the value of trust assigned by one user towards another can be significantly dependent on the recommended values by other users in the system. Thus, it is important to appropriately incorporate incoming recommendations.

Most approaches in trust-related literature deal with adversary attacks in the form of either collusion attacks (e.g., bad-mouthing, ballot-stuffing) or on-off and behavioral attacks. Adversaries that practice these kinds of attacks are normally categorized as *honest-but-curious* or *selfish* nodes, such that they attempt to reap as much benefit from the network without deviating from the specified protocol or degrading their own performance. Contrarily, *Byzantine* adversaries attempt to disrupt the performance of a network/system without any concerns over their own resource consumption [13, 115]. Trust-related literature generally deal with the former (selfish adversaries), leaving appropriate authentication and integrity mechanisms to take care of Byzantine adversaries.

In their system TrustGuard, Srivatsa et al. [200] deal with colluding and on-off adversaries, by considering a feedback-based evaluation of trust and adjusting for behavioral fluctuations on the part of the users in the network. They propose a personalized similarity measure that accounts for trust variance among the users, by taking the previous interactions among all users in the network into consideration. Sun et al. [206] consider trust as a measure of uncertainty in ad-hoc networks and attempt to facilitate efficient packet routing in ad-hoc networks with colluding adversaries. Building on recommendation trust, they propose a method to identify colluding adversaries based on their reports of packet loss on a specified processing path.

Denko et al. [56] propose a probabilistic trust management scheme in ubiquitous systems to combat against collusion and on-off attacks. In their scheme, each user compares incoming recommendation trust values against their own direct trust measurements—based on the beta distribution of previous satisfying and unsatisfying interactions. If the values vary considerably, the corresponding recommendations are considered to be falsified and the corresponding users are marked as adversaries, accordingly. Das and Islam [55] propose a feedback-based dynamic trust computation framework called SecuredTrust that overcomes oscillating adversary behavior in multi-agent systems. Similar to the approach by Srivatsa et al., they calculate trust based on historical interactions, accounting for fluctuations and decay, in addition to novel factors like satisfaction and deviation reliability. Considering the number of trust-related parameters to be assessed, their overall trust model results in increased computational delays, making it unfeasible in resource-constrained environments like D2D networks.

Chae et al. [43] focus exclusively on the management of trust in wireless networks where adversaries exercise on-off attacks. They propose a novel trust management model and redemption scheme based on predictability that distinguishes between

random unforeseen errors and purposeful malicious behaviors. By evaluating the influence of sliding windows to estimate predictability trust, the authors show how a user can find a trade-off between on-off security and performance.

2.2.2 *Discussion*

Unlike P2P systems, ad-hoc networks do not provide sufficient means to track all the messages exchanged in the network. This makes it significantly more difficult to evaluate the trustworthiness of any device, or its user. This aspect leads to the recommendation-based approach in establishing trust models in ad-hoc and D2D-based networks. While the existing efforts towards social-aware D2D networks rely on a centralized base station or cellular tower to take over the calculation of trust level of the participating devices [141, 154, 228], most approaches for trust management in ad-hoc networks either consider different schemes to evaluate trust recommendations among the involved devices [174, 206, 213] or propose game-theoretic designs [25, 169].

Considering the social aspect behind the IoT and D2D communication, we follow the path set by existing work on behavioral trust [5, 212] to evaluate the direct trust among the users. Much like other work on trust management in ad-hoc networks, we consider trust recommendation to improve existing trust values and allow for trust evolution. However, existing work on trust management does not consider the cold start problem, where there exists minimal or no previous interaction between the users involved.

Overall, there is a lack of trust-based approaches for distributed CEP in existing literature. Distributed CEP requires an increased amount of cooperation and trust among the processing devices to execute the operator graphs in a collaborative manner. Furthermore, considering the resource-constrained and mobile nature of a D2D environment, the underlying CEP system must support low-latency and battery-efficient data processing. This necessitates a light-weight computation model for trust that facilitates privacy-aware distributed CEP without adversely affecting the performance of the system, otherwise.

2.3 HUMAN RELATIONSHIPS

Building on the aspect of behavioral trust, we now turn our attention to one of the most influential factors behind it: the relationship between two people. In general, user trust evolves within a relationship with experience. A positive experience strengthens the trust in a relationship, whereas a negative experience depletes the same [172]. In particular, the type and strength of a relationship correlates positively with the amount of trust between people.

In the following, we first briefly discuss the importance of understanding human relationships, focusing on different aspects of human behavior. We then present some of the general traits of human relationships, and establish the correlation between

trust and the type and strength of relationships, before moving on to the state of the art in estimating human relationships.

Importance of Understanding Relationships

Given the advent of mobile ubiquitous computing, and with that the constant *always online* nature of its users, the problems with privacy violation have risen steadily over the past few years. User privacy constraints become increasingly important when dealing with the exchange of sensitive user context (e.g., location, activity, mood) or personal information posted on *Online Social Networks* (OSNs) like Facebook[5] and Google+[6]. Understanding human relationships and the interaction patterns within these relationships allows for the improvement and facilitation of many services for the users to combat these problems.

The characteristics of a relationship can have a strong correlation with the sharing pattern within the relationship itself [51, 78, 114, 153, 170]. For example, close friends tend to share more information among themselves than work colleagues. However, users fail to apply their privacy options on their smartphones or OSNs, appropriately [83, 142, 170, 225], leading to unnecessary privacy leaks and further unforeseen consequences, such as job dismissals and health insurance cancellations [71, 220]. Understanding human relationships allows for developing appropriate mechanisms and systems that adhere to the prevailing privacy constraints and allow the users to manage their privacy in a better manner [170]. To put in perspective, context-aware systems require the knowledge of the underlying connections (in the form of relationships) to provide the appropriate services to the users [7, 30, 114].

Another feature of the ubiquitous nature of current mobile computing are services that provide push-based notifications, so that the (mostly relevant) information is directly sent to the users, as and when any event occurs. Inopportune notifications can, however, lead to undesired interruptions, which in turn can cause unnecessary stress, mood changes, and irritability [21, 119, 131, 148], and a continual persistence of these effects can also cause a burnout [186]. Therefore, research on interruption and notification management has also increased significantly over the past 10 years [20, 137, 164, 182]. Considering that messaging applications constitute a large portion of the applications employing push-based notifications [50, 182], understanding and predicting human relationships as well as their interaction patterns allows for optimizing the time and context in which notifications (of a chosen set of contacts) should be delivered to the users. For example, in a meeting, a user may be interested in receiving notifications from his boss or certain co-workers, however notifications sent by friends or distant relatives can be deferred until after the meeting is over. By waiting for the opportune moment when a particular user is ready to receive notifications (or a particular set of notifications), one can reduce the above adverse effects [131].

5 https://www.facebook.com
6 https://plus.google.com/

General Traits of Relationships

One of the most important concepts in understanding human relationships is *social identity theory*. According to state-of-the-art research in the fields of sociology and social psychology [4, 209], social identity theory or just *identity theory* dictates that a person's sense of who they are and what their role is, depends on the group with which they interact. Basically, groups give a sense of social identity to an individual [209], and also tend to influence an individual's behavior [107].

People tend to have many social identities in general, depending on the social context they are in [151]. For example, a school teacher can be the father to his son at home, while being a teacher to his students at school. Here, the roles *father* and *teacher* describe the different *facets* or *circles* assumed by the school teacher in the different social contexts—i.e., at home and at work. The roles of two people in a relationship can vary depending on the social context, as well. For example, the school teacher can be a father to his son at home, but a teacher to his own son at school. Similarly, a work colleague who is also a friend outside working hours satisfies two different roles depending on the context. These identities thrive upon the relationship with other people and are influenced positively/negatively based on their interactions [72, 160].

Typically, the most relevant *social circles* (or *life facets*) studied in related literature are those of *family*, *work*, and *social* (friend, hobby colleague) nature [72, 142, 151, 160]. Each of these social circles expose a different behavior on the part of an individual, depending on the personality traits of the individual, and therefore, involve different interaction patterns. For example, people normally interact with work colleagues during the day, with friends in the evenings, and with family members on the weekends. Similarly, the usage of the different communication channels used by the people varies from circle to circle, e.g., some exhibiting a higher usage of calls, while others preferring message-based interactions.

Within each of these social circles, the strength of the relationship can vary from strong to weak. For example, a person can have close family members and distant relatives. Mark Granovetter [81] introduced the concept of *tie strength* to capture this sense of closeness within a relationship. Basically, *strong ties* correspond to highly-trusted relationships, e.g., between close friends and family, and is generally indicative of a stronger overlapping of the social circles. In general, people in strong ties also share more information with each other, owing to the higher trust placed on the counterpart in the relationship. *Weak ties*, on the other hand, generally corresponds to mere acquaintances and do not involve the sharing of much personal information, as far as possible. In effect, a human relationship is a two-dimensional social construct, where the social circle and the tie strength comprise the two axes, such that each point in the imaginary coordinate system corresponds to a particular tie strength within a given social circle.

Granovetter defines the tie strength between two people on the basis of four main dimensions—the amount of time spent with each other, the emotional intensity, the intimacy, and the reciprocity within the relationship [82]. These dimensions can also be extended to the study of social circles. Subsequent research on tie strength and

social cirlces in the field of sociology and human-computer interaction have led to an increased number of dimensions, including locative and temporal dependencies, relationship maintenance, number of mutual friends, socioeconomic status, political affiliation, race, and gender [125, 222]. Marsden and Campbell [132] ascertain that these dimensions cannot be measured directly, and instead suggest certain indicators or proxies for these dimensions, which allow one to measure the tie strength between two people.

Several related efforts have since proposed multiple proxies for each of these dimensions, depending on the interaction mode—online social networks, smartphone-based communication, crowdsensing, etc.—considered in their work. In the following, we discuss the state of the art in estimating the tie strength and social circle of a given relationship between two people. We particularly focus on the interaction modes and the indicators considered in each of these approaches, thus allowing us to discern the key research gaps.

2.3.1 *Predicting Tie Strength*

One of the preliminary attempts in predicting the tie strength of a human relationship was made by Marsden and Campbell [132]. After proving that the direct measurement of tie strength is difficult, they devise a model to predict the tie strength based on certain indicators like the communication intensity and frequency, among others. Through a user study and extensive empirical evaluation, they show that *closeness* or emotional intensity in a relationship is the best indicator of tie strength, but they leave fine-graded measures for closeness for future work. Furthermore, they show that contact frequency and duration are false indicators of tie strength, leading to overestimation. Furthermore, they state that relationship with family members are generally stronger than relationship with neighbors and co-workers.

Gilbert and Karahalios [78] built a model based on data obtained from Facebook to determine if a *friend* on Facebook is a weak or a strong tie, obtaining an accuracy of over 85%. 35 participants took part in a user study, and were asked to answer questionnaires with five questions pertaining to the relationship with their Facebook friends. Subsequently, the authors also extracted 74 numerical Facebook variables, such as days since last communication, educational differences, number of mutual friends, inbox positive emotion words, etc., as well as the data accessible from a user's Facebook profile to operationalize the tie strength in a machine learning system based on the obtained data from the questionnaires.

Spiliotopoulos et al. [199] applied a similar approach on estimating user tie strength based on Facebook data. They implemented the tie strength calculation functionality into Facebook, thus allowing for real-time results that can be used by the application, instantly. Their resulting model had an accuracy of around 66% in distinguishing strong and weak ties, and around 86% in distinguishing very strong and weak ties.

In their preliminary work, Wiese et al. [226] attempted to predict the tie strength between two users based on their smartphone communication history with respect to calls and SMS, achieving an accuracy of around 91%. By doing so, they could

reconfirm the assumption that frequent and intense communication does indeed indicate strong ties. In their follow-up paper [227], they further analyzed their results to establish that "a lack of communication does not necessarily indicate a weak tie". Further revelations indicate that the users nowadays prefer to communicate over other channels like instant messaging services. Also, a person may not communicate much with his/her family members over their smartphone, but may still have a close relationship with them.

Rojas et al. [178, 179] analyzed the influence of the usage of emoticons and emojis on the closeness within a relationship. In doing so, their work is the first attempt in analyzing the sentiments in chat-based communication. Based on their evaluation on the instant messenger *Skype*, they show that emoticons are a positive indicator of human sentiments and can be used to estimate the strength of a given chat relationship.

2.3.2 *Predicting the Social Circle/Life Facet*

Towards the prediction of the types of human relationships in terms of social circles or life facets, Avrahami and Hudson analyzed the influence of instant messaging (IM) communication data [12]. They undertook an analysis of PC-based IM conversations between users, and evaluated the influence of IM characteristics such as messaging rate, count, and duration on the relationship type. They achieve an accuracy of 79% in distinguishing between work and social relationships, without delving into the contents of the messages.

Eagle et al. [63] collected location and proximity information from mobile phones and analyzed their influence in determining if the relationship was a friendship or not. Using the Reality Mining dataset [62]—which includes Bluetooth-based proximity information and call log statistics—and self-reported data for the ground truth, the authors showed, among others, that friends tend to colocate more often than other relationship types. They found other correlations with co-workers based on the time of the day and location of the users.

Min et al. [142] analyze call and SMS logs on user smartphones and find the valency of their correlation with the life facet to which the contacts belong: *family*, *work*, or *social*. Going by a similar approach as proposed by Gilbert and Karahalios, the authors use supervised machine learning methods to build relationship models based on communication features from calls and SMSs as well as profile features from a user's contact list. Using user-based manual contact labeling as the ground truth, they achieve an accuracy of 87% for their models. Similarily, Reinhardt et al. [171] achieve an accuracy of 86% using communication data from call, SMS, MMS, and e-mail logs on user smartphones.

Yu et al. [234] approached this issue by assessing interpersonal relationships based on cellphone network data and in particular, by analyzing the spatio-temporal patterns based on the location and co-location information of the users. In doing so, they obtain a maximum accuracy of 73% using support vector machines as their underlying machine learning model. Bao et al. [24] present a framework called CommSense

to categorize contacts on user smartphones into the relationship types: important contacts, significant other, family, and friends. Using historical communication features from calls, SMS, along with GPS and WiFi readings in a multi-stage classification algorithm, they achieve an average accuracy of 80% for the above-mentioned categories. In terms of performance, CommSense induces an average memory load of 2.36 MB and an average CPU usage of around 7%.

Backstrom and Kleinberg [17] undertook a slightly different approach to understanding relationships. They analyzed relationship status information on Facebook and thereby, attempted to predict if a given social tie is a romantic partnership or not. They also introduce a new evaluation metric called *dispersion* to signify the amount of disconnectedness of mutual friends in a romantic relationship. Consequently, they achieved a performance accuracy of around 79% in distinguishing single and married users.

2.3.3 *Discussion*

Given that our research in this thesis focuses on D2D environments where users communicate using mobile devices like smartphones, we restrict our analysis of human relationships based on the data available on user smartphones. While phone call interactions have been proven to be a strong factor in determining the type and strength of a relationship, the usage of SMS has reduced drastically over the past few years [227]. In recent years, there has been a rise in mobile instant messaging services, such as WhatsApp[7] and Facebook Messenger[8], where users conduct quick exchange of messages and transfer of media content over the Internet. Therefore, we focus on understanding and analyzing the influence of instant messaging services on the estimation of human relationships.

Similar to the state-of-the-art approaches in related literature, we employ machine learning algorithms to extract the key characteristics from a user's communication behavior. Moving towards the concept of behavioral trust, we primarily focus on establishing the key features of the IM communication channel (alongside phone calls) in estimating the relationship, which in turn has a direct correlation with the trust within the relationship.

7 https://www.whatsapp.com/
8 https://www.messenger.com/

3 SYSTEM DESIGN

The analysis of related work in the fundamental topics behind this thesis—CEP, trust in distributed processing, and human relationships—and the identification of the main research gaps in Chapter 2 allow us to draw the key requirements of the envisioned system. Based on these conclusions, we now describe the event processing and networking models that govern the functionality of the envisioned system. Finally, we present a component-based overview of the envisioned system, highlighting the areas where our main contributions apply.

3.1 MOTIVATING SCENARIOS

To begin with, we present a set of motivating scenarios that expose the main aspects and challenges that play a key role in this thesis. Consider two smart cities, *SmartCityA* and *SmartCityB*, that are equipped with modern sensors to allow for real-time IoT processing and data analytics, supporting a context-aware environment. Walt lives in *SmartCityA* and drives every weekday to *SmartCityB*, where he works at a big IT firm. Consider the following sub-scenarios and corresponding example queries:

Scenario A: Meeting. At work, Walt wants his smartphone to adapt its profile to incoming notifications based on the situation at hand (e.g., go silent for unimportant calls during meetings). He uses a context-aware application to resolve the query if he is in a meeting or not, based on his location, the sound levels in the room, and the accelerometer readings of the surrounding people [128]. The query is resolved collaboratively on the available smartphones, but is therefore subject to the privacy constraints of the users involved (including Walt himself) as well as the resource constraints of the processing devices.

Scenario B: Traveling to/from Work. Walt uses the highway to reach his destination in the shortest time possible. On the highway, he uses an application for speed and lane assistance to help him avoid unforeseeable situations. Based on the location, speed, and direction of the vehicles around with respect to his car, he obtains information on the optimal driving speed and also, if necessary, lane changing instructions to best suit his route. Here, the query needs to be processed correctly despite the high mobility of the devices involved.

Scenario C: Fitness Route. To stay fit, Walt likes to jog in the city by considering different running profiles based on the gradient distribution along the route. He prefers routes where the air and noise pollution levels are as low as possible, yet taking a different route each time. His context-aware fitness application selects a suitable route by taking the noise and CO_2 levels due to traffic and surrounding crowds into account and estimating the best route for the jog. The query is resolved in a distributed manner using location, CO_2, and microphone readings from vehicles

and people along the route. Here, the resource constraints and random movement of the devices, as well as the privacy constraints of the users involved pose considerable challenges to resolving the query appropriately.

Scenario A accounts for low-mobility with nearly static devices, whereas Scenario B accounts for high mobility with devices moving at over $50\,km/h$. Scenario C targets situations with medium mobility, covering pedestrian movement ($2-5\,km/h$) and slow-moving city traffic ($10-50\,km/h$). With respect to privacy, each of the above scenarios involves the processing of possibly sensitive information—e.g., location and microphone readings—mainly due to their collaborative and distributed nature of processing queries. Such distributed processing allows for lower latency, reduced load on centralized servers, and better protection of one's privacy constraints. We use these three sub-scenarios to motivate our system design as well as the evaluation setup.

3.2 DESIGN CONSIDERATIONS

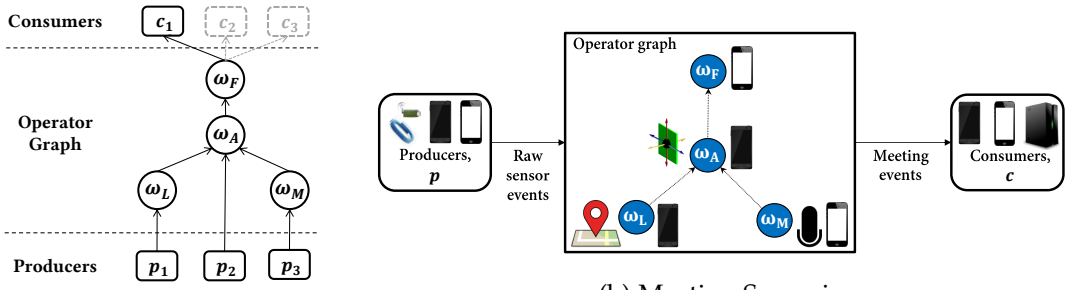
We now delve into the design considerations for the envisioned system, focusing on the concepts and assumptions that play a key role. We present a formal mathematical description of the CEP model considered in our work, including descriptions of sample operator graphs based on the scenarios presented above. Then, we present the networking model used in our envisioned system, including a brief look into the adversary model considered.

3.2.1 *CEP Model*

As mentioned in Section 2.1, a CEP system comprises a set of producers P—the sources of atomic events like sensor streams—and a set of consumers C—the sinks that are interested in the higher-level events generated by the system, like the type of meeting in Scenario A introduced above. The incoming atomic events are analyzed by a set of correlation operators Ω, which extract higher-level information by interpreting inherent patterns in the atomic events. One can model the functionality of a (distributed) CEP system with the help of an *operator graph*, $G(\Omega \cup P \cup C, E)$, which dictates the order of execution of the operators by directing the event streams E from the producers to the consumers. In turn, we can define $E \subseteq (P \cup \Omega) \times (\Omega \cup C)$.

To better understand the working of operator graphs, we now describe how the concepts of CEP can be applied in the above scenarios. We consider Scenarios A and B, so as to account for the two extremes of mobility-based scenarios.

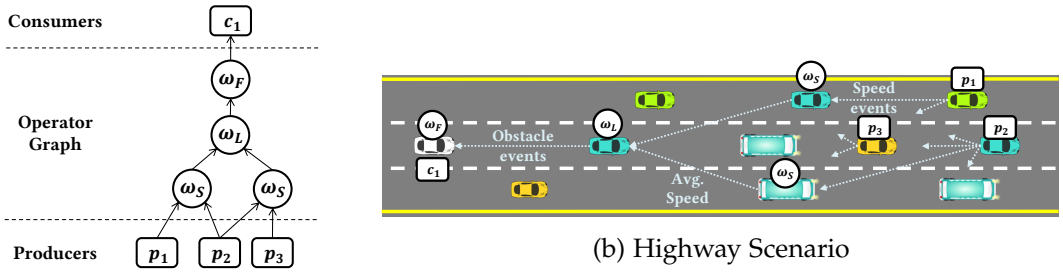
- Consider Figure 4, which illustrates the main operators used to satisfy the meeting query as part of Scenario A. Walt's phone $c_1 \in C$ is interested in knowing whether the current situation is a meeting, and if so, the type of the meeting (e.g., calm business meeting, loud informal meeting). To recognize this, the context-aware application incorporates different sound levels—obtained from smartphone microphone sensors—in the meeting room, as well as the location



(a) Sample Operator Graph

(b) Meeting Scenario

Figure 4: Sample CEP Operator Graph for Scenario A—Meeting



(a) Sample Operator Graph

(b) Highway Scenario

Figure 5: Sample CEP Operator Graph for Scenario B—Highway

and accelerometer readings of the participants [33, 128]. For a given meeting query, operator ω_M filters the microphone readings with respect to the required time frame and determines the average sound level in the given room (in decibels, dB). Operator ω_L determines the current logical location based on GPS or Wi-Fi fingerprinting [96] and filters the readings that lie outside this range. Operator ω_A then combines the output events of ω_M and ω_L, and derives the higher-level context (e.g., sitting, standing) from the movement of the users based on their accelerometer readings. Finally, operator ω_F calculates the type of the meeting based on the output events of ω_A, and sends the results to Walt's phone c_1 (and other consumers, if any).

- Let us now consider Figure 5, which illustrates the execution of a sample operator graph to detect speed and lane changes as part of Scenario B. Here, Walt's vehicle $c_1 \in C$ is interested in knowing about the average speed of the vehicles in the next $700 - 1000$ m as well as the presence of any obstacles on the route that may compel a lane change. To obtain these higher-level events, the atomic data available from the vehicles (or the corresponding smartphones), such as location, speed, and steering position, are processed using different operators. The vehicular information is transmitted from one vehicle to another using

Vehicle-to-Vehicle (V2V) communication[1], allowing for a reliable transmission range of 250 m [79]. For each road segment, operator ω_S calculates the average speed of the segment ahead over the last 30 seconds, using the information obtained from the corresponding vehicles. Further, operator ω_L predicts the possibility of an obstacle by analyzing incoming data on (average) speed and steering wheel movement. A sequence of speed change and steering wheel movement by multiple vehicles can relate to the presence of an obstacle on a given lane. The above operators can be placed on multiple devices to account for reliability in the estimated data. Finally, operator ω_F determines the recommended speed and driving lane for consumer c_1 (Walt's vehicle) based on the average speed and obstacle presence events obtained from the vehicles ahead.

Each *event* e can be described as a tuple of attribute-value pairs in the form, $e = (\langle att_1, val_1\rangle, \langle att_2, val_2\rangle, \ldots, \langle att_n, val_n\rangle)$, where event e consists of n attributes-value pairs. The contents of an event are dependent on the application at hand; typically, events contain the attributes: event type, event value, timestamp, and a source ID. Between any two devices in the system, we consider an *event stream* $(o, d) \in E$ as part of the operator graph, such that the events flow from the origin device o to the destination device d. In this context, we call o as the the predecessor or the upstream device, and d as the successor or the downstream device. Consequently, we term the event stream emerging from o as its *outgoing event stream*, and the event stream going into d as its *incoming event stream*. The attribute values of each event are kept up to date in order to facilitate further processing and in-order delivery of events.

At any given device, the incoming event stream is analyzed by an operator $\omega \in \Omega$, which is part of the operator graph G. For a given operator ω, we define the incoming event stream as $I_\omega \in E$. The operator graph and the involved operators are defined in accordance with the application at hand and the prevailing queries. Accordingly, each operator ω applies the internal correlation function f_ω on I_ω to produce a set of output (complex events) $ce \in E$ that are inserted into the outgoing event stream $O_\omega \in E$ in temporal order. We can therefore define the internal logic of an operator ω as $f_\omega : I_\omega \rightarrow O_\omega$.

3.2.2 *System Model*

In the following, we detail the characteristics of our envisioned system and state the main assumptions behind it. We base our design on the state of the art in D2D communication and in the IoT, such that users can collaboratively process contextual information based on their needs. Considering that we deal with privacy in distributed systems, we also present our model for adversaries in the system.

1 While we exclusively focus on direct communication between vehicles (or devices, in general) to avoid overloading the cellular network, one can consider placing operators on centralized instances to ensure event flow continuity when the traffic density is considerably low.

3.2.2.1 *Networking Aspects*

In our system, we consider the availability of users' smartphones as well as other sensor-fitted devices (e.g., environmental sensors to detect temperature) that provide information about the environment. Recall that our main focus lies in the IoT and the area of smart cities, where the detection and usage of user contextual information is paramount for the proper functioning of user-centric services. As shown by Wang and Peh [219], local processing directly on users' smartphones can reduce the cellular load as well as processing latency. Thus, we primarily consider users' smartphones as the devices responsible for processing the available sensor information using D2D communication. However, the proposed concepts can be extended to other distributed networked systems of autonomous computing entities using any other network architecture, as mentioned in Chapter 1.

Just as seen in the motivating scenarios, we focus on collaborative settings with distributed sources of information and distributed stakeholders for the resulting higher-level information. We consider a system model that builds on a D2D-based network spanning a set of users' smartphones. We assume that the CEP operator graphs can be executed in a distributed manner on the available devices, such that the system scales with increases in the number of producers, processing devices, and consumers. We particularly consider the mobility aspect inherent to the IoT and D2D communication, where the devices can move at varying speeds in and out of a given environment. Thus, the devices can have varying degrees of participation, joining and leaving the network in a random manner—called, device churn [202].

Given that we primarily consider users' smartphones and other environmental sensors in our system model, we deal with an ad-hoc network of devices that have a restricted set of resources (e.g., battery power, memory space) and limited visibility. Thus, we achieve scalability of the system by restricting the processing environment to the pertinent devices in the local environment. We assume that the devices keep track of the other sensing and processing devices available in the vicinity, as well as the context queries that neighboring devices are interested to process. In doing so, each user can estimate their own context depending on their application at hand, and hence obtain appropriate services, accordingly. For each context query, we consider a *leader* device that is chosen based on a consensus beforehand amongst all pertinent devices—e.g., based on the available battery level, network connectivity, or mobility pattern. The leader initiates the deployment of the corresponding operator graph, based on the context query at hand. During the placement of CEP operators on available devices, we account for the privacy constraints of the leader as well as the participating devices (neighboring collaborators) to achieve privacy-aware event dissemination in the distributed CEP system.

To account for reliability, we primarily focus on the migration of the CEP operators and the involved events upon changes in the system—because of device disappearance (e.g., due to mobility) or failure (e.g., due to resource constraints). We assume that suitable (backup) devices are available for migration that can take over the working of the (dangling) operator(s) on previously active devices. We utilize available reliable communication primitives in the underlying network to overcome

any network flaws, such as link fluctuations and routing issues, so that the messages are delivered completely, in time, and with maximum possible throughput.

3.2.2.2 *Adversary Model*

Distributed systems can be subjected to a wide range of security/privacy attacks. In the context of the IoT and D2D communication, the possibility of attacks becomes even more pronounced, since we deal with wireless ad-hoc networks [230]. Such attacks can range from eavesdropping and man-in-the-middle attacks (leading to, e.g., authentication violation, routing traps), to DoS attacks (causing, e.g., resource depletion, system breakdown), as well as Byzantine and alteration attacks (through, e.g., arbitrary falsified information) [120].

In this thesis, we primarily focus on preventing and overcoming attacks by *honest-but-curious* adversaries, which mainly satisfy the properties of alteration and Byzantine attacks, without adversely affecting their own resources. In our case, on the one hand, the adversaries perform their delegated tasks—the execution of the CEP operators by disseminating the events to the intended recipients—by conforming to the corresponding operator graph (i.e., *honest*). On the other hand, they can resort to malicious behavior by adversely affecting the system functionality to obtain as much sensitive information as possible from the *benign* users in the environment (i.e., *curious*). An analysis of all possible attacks in one such environment goes beyond the scope of this work. The other security/privacy attacks can be combated using appropriate measures, as established in related literature [68, 120, 155].

In our work, being honest but curious, the adversaries attempt to manipulate the system functionality through two types of attacks: *collusion* attacks [200] and *on-off* [43] attacks. In collusion attacks, a set of adversaries cooperate with each other to influence the event dissemination in the system, such that they attempt to redirect the events from the benign users towards themselves. Our concept for a privacy-aware implementation of distributed CEP is based on the inherent trust level between the users as well as their recommendations, as briefly pointed out in Section 1.2. Therefore, the adversaries in our system attempt to improve their trust level in the eyes of the benign users, so that they may receive the (possibly sensitive) events. In turn, they attempt to manipulate their trust recommendations to improve their own reputation in the system. They do so by either increasing the recommended trust levels of their fellow colluding adversaries (i.e., ballot-stuffing) or decreasing the recommended trust levels of the benign users (i.e., bad-mouthing). In on-off attacks, the adversaries pretend to behave benignly intermittently for short intervals of time—improving their reputation in the eyes of the benign users as they do so—before turning malicious. When malicious, the adversaries practice collusion with other adversaries, as described.

3.3 COMPONENT OVERVIEW

Figure 6 presents an illustration of the main components involved in the privacy-aware and reliable CEP system proposed in this thesis. As discussed in Section 3.2.2.1,

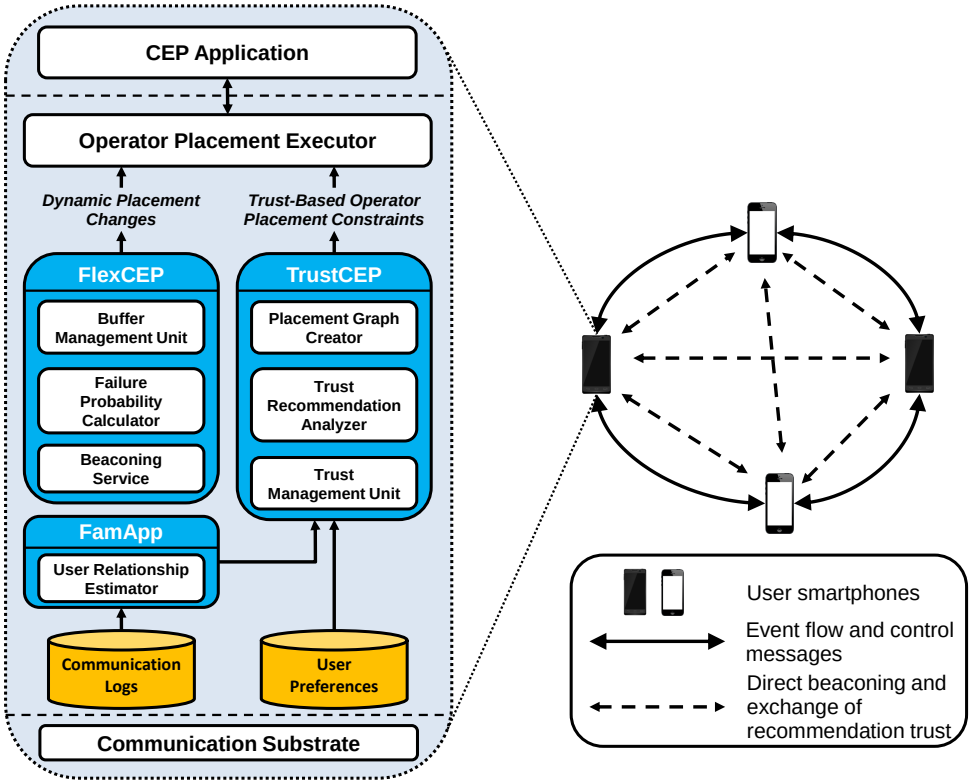


Figure 6: Component-based Depiction of the Proposed System

we consider a D2D-based network of user smartphones that interact directly with each other and process sensor-based information in a distributed and collaborative manner. Consequently, each smartphone works as an individual CEP device to enable the execution of CEP operator graphs in collaboration with the other CEP devices in the network.

In our proposed system, we assume that the query to be processed is provided by the CEP application at hand. For example, a context-aware telephony application used in Scenario A resolves a query to determine if a meeting scenario is taking place. Accordingly, a similar CEP operator graph to that shown in Figure 4 is deployed and executed on the available devices. We also assume that there exists a reliable communication substrate underlying our implementation to facilitate data exchange between the CEP devices, i.e., user smartphones. In turn, we implement our system between the communication substrate and the CEP application.

Our proposed CEP system consists of three main components, in accordance with the three main contributions of our thesis, as presented in Section 1.2.

(i) The first component called FamApp deals with the estimation of user relationships based on their communication-based interactions. In particular, we propose and evaluate a method to analyze the historical communication data on synchronous and asynchronous communication channels, and determine their

correlation with the relationship type (social circle) and strength (tie strength) of the users in question. To this end, we developed a smartphone-based application called FamApp that extracts the required features from the communication logs and classifies them into the different social circles (e.g., friends, family members) and tie strength categories (e.g., strong and weak). Effectively, the resulting analysis allows us to determine the key features that help in distinguishing between different relationship types, and thus derive the trust relations between the users.

(ii) The results of the FamApp component act as the input to the TrustCEP component, which is responsible for the privacy-aware placement and execution of CEP operator graphs in accordance with the trust relations between the users. Using this information, the *Trust Management Unit* determines the trust vector of the smartphone user with respect to the other users. In doing so, it also weighs in possible user preferences, which include—(i) the general character traits of the users in employing their trust level towards the other users [221], and (ii) their privacy preferences with respect to the events to be disseminated. The *Trust Recommendation Analyzer* is responsible for determining the credibility of incoming trust recommendations and to discover (possible) adversaries in the environment. Based on the derived trust levels, TrustCEP prepares the placement graph for the CEP operator graph at hand. Note that, as explained above, the *leader* device takes over the responsibility of deploying the operator graph, based on its trust level vectors as well as the recommendation vectors received from the other devices.

(iii) Finally, the FlexCEP component accounts for the reliable execution of CEP operator graphs, even in highly dynamic user environments. These dynamic changes can occur due to changes in the trust relations, as dictated by the TrustCEP component, or due to device movement or malfunction. To this end, each smartphone employs a beaconing service to exchange status information about themselves with the other user smartphones. The collected status information beacons are then used to determine the probability of *failure* of the other devices. The knowledge of failure probability allows us to flexibly migrate CEP operators running on failure-prone devices to other capable devices. The *Buffer Management Unit* accounts for the adaptive management of recovery buffers on the CEP devices, so as to extract the required operator state as well as reduce the required buffer space during migration.

The combination of these three main components contributes to the proposed system for privacy-aware and reliable CEP in dynamic distributed environments.

4

ESTIMATING USER RELATIONSHIPS

Building a privacy-aware distributed processing system based on user trust first requires the establishment of trust relations between the devices in the system. In our work, we model the trust between devices as an extension of the trust between their users, where we mainly concentrate on the influence of user trust on the amount of information shared among themselves. A key factor that dictates the trust level between users is their relationship as well as their interaction patterns [133, 227]. The type and strength in a given human relationship plays a key role in establishing the type and intensity of interpersonal communication—e.g., phone conversations, e-mail correspondence, personal meetings [63, 142, 227]—which in turn helps in determining the amount of trust between users.

In this chapter, we show our results on how we can determine the type and strength of human relationships by analyzing their communication patterns. We first provide a brief explanation of the *dimensions* of human relationships, followed by an overview of our approach based on the background presented in Section 2.3. We then detail our analysis of human relationships based on communication patterns on users' smartphones, before presenting the key influencing factors for the estimation of human relationships. The obtained results act as the input to the next system component, TrustCEP, where we establish the trust relations between the devices.

4.1 UNDERSTANDING RELATIONSHIP DIMENSIONS

Recall from Section 2.3 that human relationships can be perceived as a two-dimensional social construct, comprising the type of relationship—i.e., the social circle(s) in which the relationship falls—and the strength of the relationship—i.e., the tie strength. We term this two-dimensional construct as the *familiarity* level of the relationship. The familiarity level plays an important role in establishing how users interact with each other, and therefore, how users share information with each other. For example, in Scenario A as described in Section 3.1, Walt will be willing to share his location information with his work colleagues, but he may not be willing to share the same information with his external clients. On the other hand, he may be less inclined on sharing his location information with his colleagues in Scenario C, considering that the action takes place outside working hours. Existing work on human-computer interaction has shown that users do not use the privacy settings on their smartphones or on OSNs appropriately, leading to inadvertent privacy leaks [220]. Understanding human relationships, and in particular, estimating the social circle and the tie strength of a given relationship, is a necessary step towards preserving privacy during information exchange among users.

There are a multitude of social circles present nowadays in human society, given that each individual can assume multiple identities in the different relationships [160, 209]. In our work, we restrict the different social circles to the following—*friend*, *family*, *work*, *significant other*, *hobby*, and *others*, where, the *others* mainly include people with whom one does not communicate very often or not at all. We formed this list based on existing work in the field privacy management and sharing patterns [51, 114, 153]. A person can also assume multiple roles with another person, depending on their individual activities in daily life. Each of these roles, in turn, is reflected in a distinctive pattern in their communication behavior [50, 227].

Recall from Section 2.3, the tie strength is characterized by four main dimensions of the relationship—time spent together, emotional intensity, intimacy, and reciprocity [82]. A standard procedure for measuring each of these theoretical dimensions is not very practical, primarily since the implications of these dimensions depend on the corresponding interaction sequence—phone call, e-mail, in person, etc. In our work, we consider the inference of these dimensions based on communication patterns available on user smartphones.

In order to approximate these theoretical dimensions, we consider so-called *indicators* that act as a proxy for measuring each dimension [132]. Typical indicators include the duration of communication (e.g., a phone call) and the frequency of interaction on a particular communication channel [227]. Depending on the type of interaction channel considered, other relevant indicators emerge, such as the diversity of topics discussed and number of common friends or activities, that have been proven to be significant indicators in recent social network research [17, 78].

Additional indicators that depict the temporal and locative dependencies in the communication patterns between people help in distinguishing among the social circles to which they belong, as described in Section 2.3. The communication intensity and frequency is also influenced by the amount of effort one puts into the relationship, termed as relationship maintenance indicator [177]. This shows how mutual and reciprocative the relationship is.

In accordance with sociological references [82, 132], we propose the following set of indicators to estimate the type and strength of human relationships—communication intensity and frequency ($\mathcal{I}$); temporal dependency ($\mathcal{T}$); locative dependency ($\mathcal{L}$); and the aforementioned relationship maintenance ($\mathcal{M}$). The two-dimensional familiarity of a relationship ($\mathcal{R}$), which in turn comprises the social circle ($\mathcal{C}$) and the tie strength ($\mathcal{S}$), can therefore be described as a (typically non-linear) function of the above indicators:

$$f_{\mathcal{R}} : (w_c, \mathcal{I}, \mathcal{T}, \mathcal{L}, \mathcal{U}, \mathcal{M}) \rightarrow \{\mathcal{C}, \mathcal{S}\} \tag{1}$$

where, w_c is a set of weighting/correlation factors for each of the indicators, such that the resulting function determines the social circle and the tie strength of the relationship at hand. In order to obtain a cogent estimate of the familiarity of a given relationship, the above indicators must be weighed in appropriately. The following sections delve deeper into the extraction of the required indicators based on smartphone data, and the results obtained based on their analysis.

4.2 ANALYZING USER RELATIONSHIPS USING FAMAPP

The main objective of the relationship estimation component is to understand how inter-user interactions, in particular over communication channels, can be used to infer the relationship between users. In our work, we employ *supervised machine learning algorithms* [117] to combine the indicators obtained from inter-user communications in accordance to their relevance, based on a set of labeled data provided by the users. The former aspect of our approach entails the extraction of the indicators of a user relationship from the available communication history between two users, called *data extraction*. The latter aspect of our approach involves the collection of the *ground truth*[1], so that we can correlate the communication indicators to the corresponding social circle and tie strength of a given relationship.

For the purpose of extracting the required indicators as well as the ground truth, we developed an Android-based[2] smartphone application called FAMAPP[3] (short for ***Fam**iliarity **App**lication*). With the help of this application, we can collect the relevant data from user smartphones. Using FAMAPP, we put forth the following contributions towards estimating user relationships:

C4.1 We devise a method to extract the relevant indicators from user smartphones, pertaining to the communication channels (calls and messages) (see Section 4.2.1). We detail the steps taken towards the extraction of the indicators, including their significance towards the relationship dimensions mentioned in Section 4.1.

C4.2 We describe the user interface of FAMAPP as well as the details of our user studies to obtain the ground truth required for our method based on supervised machine learning (see Section 4.2.2). The obtained datasets can be utilized for further research in the field of human-computer interaction.

C4.3 Finally, we perform a thorough analysis on the obtained datasets using supervised machine learning algorithms to understand the influence of the extracted communication-based indicators on the estimation of the social circle and tie strength of a relationship (see Section 4.3).

4.2.1 *Data Extraction*

Smartphones are the most predominant mobile devices used by people nowadays. According to a recent survey, smartphones have shown a steady increase in the number of users over the past few years and are expected to continue increasing in the future [66]. Based on this insight, we restrict the communication data for extraction

1 In machine learning jargon, the *ground truth* is the information obtained by direct observation (e.g., expert opinion, user input), and not based on inference. It assists in estimating the accuracy of a classification algorithm in supervised machine learning. However, it can be error-prone because of the subjective nature of its measurement.

2 https://www.android.com/

3 https://play.google.com/store/apps/details?id=com.solin.slc

to the data available or accessible on user smartphones. In particular, we focus on understanding how the communication behavior on *Instant Messaging* (IM) applications is influenced by a relationship. There has been a sudden proliferation of IM-based services over the past few years, with a wide range of applications coming to the fore—e.g., *WhatsApp*[4], *Facebook Messenger*[5], *Google Hangouts*[6], *Threema*[7]. We extract the necessary indicators from IM communication data on user smartphones, pertaining to the relationship indicators introduced in Equation (1). In addition to the IM data, we also extract relevant relationship indicators from call and SMS logs, as done by other related work on human relationship estimation [142, 156, 227]. In doing so, we want to validate the general assumption that call intensity and duration have a positive correlation with higher values of tie strength.

In the following, we present the steps taken to set up our smartphone-based application FamApp, especially in extracting the required indicators based on IM communication data. We first discuss the issues in extracting the required information from user smartphones, before presenting the list of indicators extracted, corresponding to the different dimensions of user relationships. The extracted indicators act as the *feature set* for the supervised machine learning algorithms.

4.2.1.1 *Using Smartphones as Information Sources for Data Extraction*

WhatsApp has been the most dominant mobile IM application service in recent years. A recent study showed that the number of messages exchanged over WhatsApp already increased to 64 million in April 2014, with an exponential increase each year [223]. Even the number of active users over the past 5 years has increased exponentially, recently having crossed the 1 billion mark [224]. Given the almost unanimous popularity of WhatsApp, we incorporate it as one of the IM applications for data extraction. In addition to WhatsApp, we include the secure messaging service called *Threema*, which has received a lot of popularity in Germany due to its security and usability features. While we choose these two IM applications, it should be noted here that the principles of our data extraction module can be applied to other IM applications, too.

One major problem with the extraction of indicators from the IM applications is the lack of suitable tools for retrieving communication histories [227]. Most existing IM applications, including the ones mentioned above, have proprietary closed-door models behind their *Application Programming Interface* (API), which does not allow external applications to access their internal content. In order to circumvent this problem, we decided to access the status bar notifications that are used to notify users of any incoming messages. These notifications include all the necessary meta-information about an incoming message (e.g., sender name and phone number), as well as—at least in WhatsApp—the message content itself. This allows our data extraction module to save the required indicators based on each notification as a log

4 https://www.whatsapp.com/
5 https://www.messenger.com/
6 https://hangouts.google.com
7 https://threema.ch/en

entry. Of course, this does not provide the entire picture of a conversation, given that we cannot track the outgoing messages. However, as we show later, we can still draw reasonable interpretations based on these indicators with respect to a relationship.

In order to obtain the locative indicators mentioned in Equation (1), we extract the phone location based on the cell tower IDs, to which the smartphone is connected at a given point in time. By allowing the users to tag the cell tower IDs with the corresponding logical locations (e.g., home, work), we allow for a battery-saving and privacy-preserving alternative to GPS-based sensing. Furthermore, many users tend to switch off their GPS sensing services on their smartphones due to the said problems concerning privacy and battery drain [161]. Using the stored information for future reference, we can infer the users' logical location whenever they enter the stored cell tower zones. We append this information about the users' location to the extracted indicators from the communication channels.

One of the key characteristics of messaging applications is the option to introduce emotions into the messages with the help of emoticons/emojis. The analysis of emoticons in text messages as well as their correlation with the actual human emotions has been the subject of research in recent years [178, 179, 218]. Furthermore, the emotion and opinions behind the message can also be interpreted by analyzing the usage of words within the message. We analyze the influence of message contents on estimating human relationships by extracting the required indicators based on the message sentiment and the usage of different types of emoticons.

To evaluate the usage of emoticons in messages, we group the most common emoticons into three main categories—love/affection, negative/sad, and general—based on the lists established by the Unicode Consortium[8]. In total, we consider 8 different love/affection emoticons, 17 negative emoticons, and 37 general emoticons, which express neutral and pensive emotions. We analyze the impact of the usage of emoticons by counting the number of emoticons from each group used in each message.

We evaluate the sentiments behind message contents by applying the concepts of *opinion mining* in the field of *Natural Language Processing* (NLP) [109]. Generally speaking, opinion mining aims at the identification of the overall contextual polarity of a document by analyzing the opinions of the speaker/writer. In the context of IM and SMS messages, we classify whether a certain message sent/received has a positive or negative connotation. To do so, we use two well-established libraries for opinion mining—SentiWordNet [16] for messages in the English language; and SentiWS [173] for German messages. We adapt each of these libraries to return the lexical sentiment score, which indicates whether a message is positive or negative. This allows us to include a lighter version of the opinion mining engine in the smartphone application, without overloading the processor RAM on user smartphones.

4.2.1.2 *Feature Extraction for Supervised Machine Learning*

Using the above sources of information on smartphones, we apply supervised machine learning algorithms on the extracted indicators. To obtain the required indicators—

8 https://unicode.org/emoji/charts/full-emoji-list.html

called features in the context of machine learning—we extract data from the communication activity on smartphones. Overall, we define a set of 127 features for the supervised machine learning approach: 62 features based on messaging interaction on IM and SMS applications, 56 features from call logs, and the remaining 9 features as a combination of both the call-based and message-based interaction. These features correspond to the indicators of human relationships, introduced in Equation (1). The complete list of features used in our work is presented in Table 1. In the following, we explain the importance of selecting these features from each communication channel with respect to the relationship dimensions introduced in Section 4.1.

Messaging Features. We consider incoming and outgoing messages in the SMS channel, but only incoming messages in the IM channel, due to the limitations mentioned before. With respect to the IM channel, we analyze the number of messages exchanged as well as their average length, which represent the features for message intensity and frequency. It has been proven by existing work in social network research that SMS interaction lacks temporal dependencies, given its asynchronous nature of interaction [29]. Hence, we try to validate this observation on IM data by measuring the time taken to respond to a message, thus accounting for temporal dependence. However, given the lack of appropriate APIs to access this information, we instead measure the time taken by a user to react to received messages. This is done by measuring the time taken to open the corresponding IM application after a message arrived. We thus calculate the percentage of IM messages *read* within 3 and 10 minutes.

Additionally, we include the analysis of emotion and sentiment within message content in both SMS and IM, to account for channel usage indicators. We measure the number of emoticons used per message as well as the percentage of messages with emoticons. Furthermore, we calculate the percentage of general, positive, and negative emoticons used in messages. We also calculate the percentage of messages with positive/negative sentiment, as well as the standard deviation of message sentiment for IM, based on sentiment analysis libraries mentioned before. Given the steady decline in popularity with respect to SMS interaction [227], we restrict its analysis to the study of emoticons and message sentiment for both incoming and outgoing messages.

Call Features. For the call-based features, we mainly rely on existing work in the field of relationship estimation [142, 171, 227]. The main features include the total number of calls and total call duration, which account for the intensity and frequency of calls. To measure the temporal dependency, we distinguish call activity on the weekdays to that on the weekends, and also differentiate between calls on Fridays, Saturdays, and Sundays, to deal with the different call practices in the social circles considered. We account for locative dependencies by exploring call activity at the logical locations *home*, *work*, and *university*. Furthermore, we measure the superiority of call-based interaction over the other channels by including the ratio of call-to-overall communication. We also incorporate the number of lengthy calls by accounting for all outgoing calls where the call duration exceeded 10 minutes, to measure communication intensity.

Table 1: The Extracted Communication Features and Their Corresponding Dimensions

Comm. Channel	**Feature Variables**	**Indicator Set [s. Equation (1)]**
Messages (*Phase I*: 16) (*Phase II*: 62)	No. and avg. length IM [incoming]/SMS [incoming, outgoing]	*Communication intensity and frequency,* $\mathcal{I}$
	% IM read within [3, 10] mins	*Temporal dependency,* $\mathcal{T}$
	Avg. number of emoticons per IM/SMS; % IM/SMS with emoticons; SMS [incoming, outgoing] avg. [general, pos., neg.] emoticons; SMS [avg., σ] sentiment score [DE, EN]; IM avg. [general, pos., neg.] emoticons; IM [avg., σ] sentiment score [DE, EN]	*Channel usage,* $\mathcal{U}$
Calls (*Phase I*: 56) (*Phase II*: None)	Number; duration; days with calls $\geqslant 2$; No. and max. duration of incoming calls; σ outgoing call length; No. of lengthy calls (> 10 mins)	*Communication intensity and frequency,* $\mathcal{I}$
	% Calls and % call duration on [Friday, Saturday, Sunday, weekdays, before noon]; % long calls on weekdays	*Temporal dependency,* $\mathcal{T}$
	Number, duration, and % at [home, work, university]	*Locative dependency,* $\mathcal{L}$
	% Calls to all communication [overall, weekdays, Fridays, Sundays]	*Channel usage,* $\mathcal{U}$
Combination (*Phase I*: 8) (*Phase II*: None)	No. of all communication events; No. of days of communication	*Communication intensity and frequency,* $\mathcal{I}$
	% Calls, SMS, and IM on Saturday	*Temporal dependency,* $\mathcal{T}$
	Number of channels used	*Channel usage,* $\mathcal{U}$
All of the above in the past [14 days, overall]; days since last communication		*Relationship maintenance,* $\mathcal{M}$

Combinatorial Features. There exists a fundamental dependence between the different communication channels, given that an event in one channel can result in an event in another. For example, some text messages may require an immediate phone call from the receiver, or vice versa. To account for these cases, we include combinatorial features, such as the total number of communication events, and the total number of channels used. We also include certain features to account for temporal dependencies, like the percentage of communication on the weekends.

In order to account for the fourth dimension, i.e., relationship maintenance, we measure all the above features in two time frames—over a shorter period of the past 14 days, as well as a longer time period of 90+ days, depending on the available data. Also, we include a feature for the number of days since the previous communication event, which indicates how often people communicate with each other.

4.2.2 *User Study*

We conducted two separate user studies to evaluate the influence of IM patterns on the estimation of human relationships. In the first user study, we mainly focus on comparing the characteristics of IM usage with the usage of calls and SMS, and analyzing their respective contributions to estimating the type and strength of human relationships. In the second user study, we delve further into the characteristics of messaging services, mainly laying our focus on the contents of the messages exchanged and understanding their influence on the relationship.

4.2.2.1 *Phase I*

We carried out the first user study (*Phase I*) over a period of 4 weeks in July 2015. In order to obtain participants for our study, we advertized FAMAPP among fellow work colleagues as well as students at the university. All users were requested to install the application on their phone and provide the application access to their contact list, call and SMS logs, as well as the notifications on the status bar. In turn, of the 127 features presented in Table 1, we considered 80 features in *Phase I*, which includes all call-based and combinatorial features but only the 16 basic IM/SMS features, not including the features pertaining to message content (except basic emoticon features).

Considering that our approach is based on supervised machine learning, we collect the ground truth data from the users, by asking them to assess a selection of persons from their contact list. In order to account for all types of contacts—i.e., strong and weak relationships—we chose a balanced set based on the total duration of calls with the respective contact persons, given the proven correlation between call duration and tie strength [156, 227]. Android-based smartphones normally maintain logs of the last 500 calls and typically 200 SMS messages per contact [142]. Hence, upon installation of the application, the initial set of suggested contacts for assessment is based on the median of the aggregated call duration over the past 90 days from the call log. The remainder of the assessed contacts are suggested based on the user's interaction patterns on the observed communication channels. Consequently, we mainly account for contacts that have recently been engaged in a conversation.

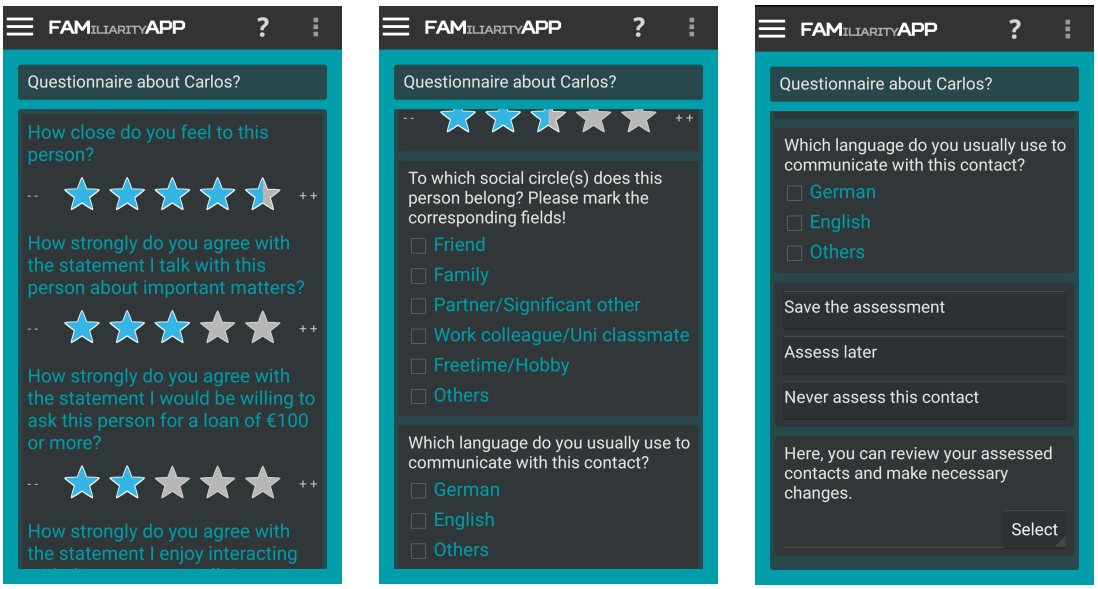


Figure 7: The User Interface for Contact Assessment in FamApp

The collected ground truth is based on the responses to the questionnaires pertaining to each suggested contact. In each questionnaire, the users are requested to allocate the suggested contact to the corresponding social circle(s). We allow the users to choose more than one social circle, so as to account for contacts that satisfy multiple roles. Furthermore, we also allow the users to skip contacts, which provides the means to account for uncertain or irrelevant contacts—e.g., landline numbers, insurance companies, answering machine services—that do not exhibit a typical communication pattern. Alternatively, the users can classify these contacts as *others*.

In addition to that, the users are also asked to answer a set of four questions pertaining to the tie strength with the suggested contact. The responses are collected using a 10-step Likert scale from 0 to 100, in steps on 10. The four questions are based on previous work in the field of sociology and human-computer interaction [78, 130, 227]:

1. How close do you feel to this person?
2. How strongly do you agree with the statement "*I talk with this person about important matters*"?
3. How strongly do you agree with the statement "*I would be willing to ask this person for a loan of EUR 100 or more*"?
4. How strongly do you agree with the statement "*I enjoy interacting with this person socially*"?

The questions account for the amount of social interaction, willingness to borrow money, and general closeness (emotional intensity) between the user and the

suggested contact. The ground truth for the corresponding tie strength is given by the linear sum of the responses to each of the four questions. Figure 7 provides an overview of the contact assessment page on FamApp, showing the questions for tie strength analysis as well as the check-boxes for the social circles.

During *Phase I*, in order to preserve user privacy and increase user acceptance, we did not collect any demographical data from the participants. Furthermore, all the information submitted by the participants were saved on their respective smartphones for later retrieval. Each day, the saved datasets were uploaded securely to an external server, where they were saved using a pseudonym based on the Android ID of their smartphones.

4.2.2.2 *Phase II*

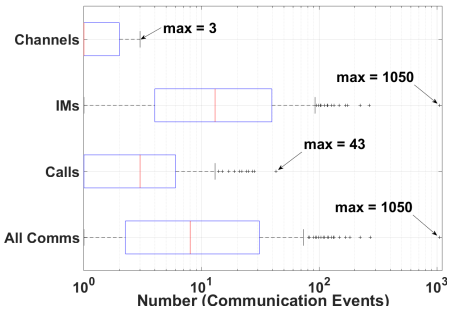
For the second user study (*Phase II*), we modified FamApp slightly to mainly account for the contents of the messages exchanged using IM and SMS. In particular, we accounted for the usage of emoticons and the sentiments in the messages exchanged, as introduced in Section 4.2.1. In doing so, we mainly chose the 62 features based on IM and SMS communication, as well as the combinatorial features based on messages, as presented in Table 1.

We carried out *Phase II* over a longer time period of 4 months from March to June 2016. Instead of entirely restricting our participants to fellow work colleagues and students, we advertized the application on the Web and in other universities in Germany. Furthermore, as an incentive for the participants, we provided them the opportunity to win gift vouchers of up to 50€ from an online shopping website, based on the number of contacts they assessed as part of the study.

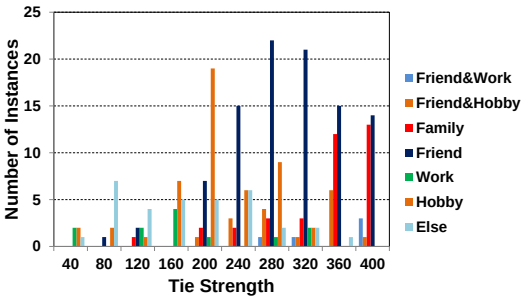
Similar to *Phase I*, the participants were requested to install our application and provide access to the above mentioned smartphone logs and services. In addition to the tie strength and social circle related questions in the previous questionnaire, we also included a question that pertains the language used for communication with the suggested contacts. Considering that the application is based in Germany, we mainly consider the languages of German and English. This allows us to apply appropriate algorithms to analyze the sentiments behind the messages. Furthermore, this allows us to obtain an overview of the demographic information of the participants.

4.3 EVALUATION AND RESULTS

In this section, we provide an overview of the collected datasets in each user study and detail our approach for evaluating the same with the help of supervised machine learning techniques. We will first present the results of the evaluation of the datasets obtained in *Phase I*, before shifting our focus to the analysis of message contents in *Phase II*.



(a) Distribution of the Main Extracted Communication Features per Participant



(b) Distribution of the Tie Strength over Different Relationship Types

Figure 8: Overview of the Data Extracted in *Phase I*

4.3.1 *Results of* Phase I*: Analyzing Call and Messaging Indicators*

In *Phase I*, we focused on evaluating call and messaging indicators to estimate the type and strength in a human relationship, and to understand the key contributing factors that characterize each relationship. A total of twelve users participated in our study in *Phase I*. However, two of those users used the application for fewer than ten days. In order to avoid the possibility of falsifying the models generated by the machine learning algorithms, we decided to eliminate the datasets provided by the corresponding users. Furthermore, two other users returned the default values (50) for the questions pertaining to tie strength for all the suggested contacts. We assumed that these values do not reflect the true tie strength, and therefore, decided to eliminate their datasets, as well. After these changes, we obtained a final dataset by eight users, comprising a total of 249 instances, where each user assessed an average of 31 contacts ($min = 4, max = 58, \sigma = 18.96$).

Table 2 shows the distribution of the obtained dataset across the different social circles considered in our work. We notice that most contacts were classified as *friends*; there are considerably fewer number of contacts that were assessed as *family members* or *work colleagues*; only 2 contacts were assessed as the users' *significant other*. The latter attributes to the low number of users that took part in our study. Considering the low number of instances of *significant others* as well as the multifaceted nature of the social circle *others*, we decided to disregard these two social circles in the following evaluation.

Figure 8a shows the distribution of the communication events per user across the two main channels—calls and IM—in *Phase I*. We notice that the number of instant messages received is considerably higher, with a median of 13 messages received compared to a median of 3 calls made/taken during the study period. This shows a clear trend towards IM interaction. Overall, we extracted a total of 7191 communication events (calls, SMS, and IM), of which there were 680 calls, 246 SMS messages, and 6265 IM messages.

Table 2: The Social Circle and Tie Strength Distribution in *Phase I*

All Instances	Social Circles						Tie Strength (StrongWeak)		Tie Strength (VerystrongWeak)	
	Friend	Family	Work	S.O.	Hobby	Others	Strong	Weak	Very Strong	Weak
249	120	36	17	2	71	38	72	177	38	211

Table 3: Channel Usage Based on Social Circle (*Phase I*)

		All	Friend	Family	Work	Hobby
Original Set		249	120	36	17	71
***Subset 1*: Using Calls**	⇒	139	69	30	12	16
	Linear decrease	139	66.99	20.10	9.49	39.63
	% Rel. change	–	+3.00	+49.28	+26.45	-59.63
***Subset 2*: Using IM**	⇒	174	93	19	6	61
	Linear decrease	174	83.86	25.16	11.88	49.61
	% Rel. change	–	+10.91	-24.47	-49.49	+22.95

Before we proceed with the analysis of the extracted dataset, let us analyze the usage of the communication channels in the different social circles in order to draw an initial understanding of different usage patterns. To this end, we divided the dataset into two subsets: *Subset 1* with instances[9] with at least one call event; and *Subset 2* with instances with at least one IM message. Table 3 shows the distribution of the two subsets across the social circles considered. The row values '*Linear decrease*' show the number of instances in each social circle if there were a linear decrease with respect to the decrease in the number of instances in the overall dataset. For example, the overall dataset decreases by 44% (from 249 to 139 instances) for *Subset 1* if we consider all contacts that at least had one call event. If we consider such a linear decrease in each social circle, we observe that, e.g., the number of *family* instances would be at around 20. The row values '*% Rel. change*' show the relative deviation of the true number of instances in each social circle in each subset to the number of instances by linear decrease.

We observe that the number of instances of *family* and *work* is considerably higher than the predicted number based on a linear decrease (+49.28% and +26.45%, respectively). This shows a clear preference for call-based interaction in these two social circles. Likewise, we also observe that the *friend* and *hobby* contacts show a distinct tendency towards IM-based interaction, given the relative increase by 10.91% and 22.95% in the true number of instances in these two social circles, respectively. Furthermore, we can discern that *family* and *work* instances tend to steer clear of IM-based interactions, given their significantly negative deviation from a linear decrease (-24.47% and -49.49%, respectively). Also, *hobby* instances do not show any notable inclination towards call-based interactions (-59.63%), which attributes in general to the usage of short message based interaction in this social circle.

Figure 8b provides an overview of the tie strength distribution over all instances for the different social circles. As mentioned earlier, the tie strength ground truth is based on the linear sum of the responses to the questions in the app-based questionnaires, such that a higher value denotes a higher tie strength. We observe that there is a shift towards the higher regions of tie strength ($mean = 250.6$, $\sigma = 97.9$). This indicates that people generally seem to save contacts in their smartphone contact lists with whom they have an above-average tie strength. We also can discern that the strong contacts primarily fall in the social circles of *family* and *friend*, whereas exclusively *work* and *hobby* instances tend to have a weaker tie strength. Interestingly, we observe that the *work* and *hobby* instances who have also been marked as *friend* exhibit a much higher tie strength, with the contacts who belong to both *work* and *friend* having a mean tie strength of 360, and that of *hobby* and *friend* lying at 307.5.

Given these findings, we include two additional datasets in the ensuing evaluation—dataset *OnlyC* comprises only the data belonging to the 50 exclusively call-based features, whereas dataset *OnlyM* consists of the data belonging to the 16 exclusively messaging features, i.e., both IM-based and SMS-based features (see Table 2). In doing so, our main focus lies in understanding the influence of each of these com-

9 An *instance* is an assessed contact person in the app-based questionnaire.

munication channels on the estimation of the relationships. In turn, we also remove the combinatorial features from the two datasets.

4.3.1.1 *Dataset Preparation*

Before we can proceed with the analysis of the obtained dataset for the estimation of social circles and tie strength, we have to prepare the dataset accordingly. With respect to the social circle, as seen in Table 2, there is clearly a significantly larger number of *friend* instances compared to the other social circles, leading to strongly imbalanced datasets. Given the low number of representative instances for the other social circles—e.g., *family* or *work*—the social circle classifier would not be able to classify them properly. One approach to overcome imbalanced datasets is resampling [142, 171], where the instances in the original dataset are randomly selected (i.e., resampled), such that all social circles (or *classes* in the machine learning jargon) in the dataset are equally represented (or at least, have a sufficient number of representative instances).

For the tie strength analysis, we consider a nominal approach and first group the assessed contacts into *strong* and *weak* instances. We consider a score of 320 out of 400 (80%) as the threshold for the *strong* instances. In turn, we obtain a total of 72 instances for *strong* and 177 for *weak* (see Table 2). We term the corresponding dataset as *StrongWeak*. In accordance with related work on tie strength analysis [199, 227], we also consider an additional dataset to distinguish the highly strong ties from the rest of the contacts. To this end, we divide the contacts into two categories—*very strong* and *weak*—by setting the threshold at 90% (360 out of 400), resulting in 38 very strong and 211 weak instances, thus forming the *VerystrongWeak* dataset (see Table 2). Due to the imbalanced nature of these datasets, we employ a similar resampling approach as described above to obtain better classifier models.

Two of the main issues in the generation of models based on machine learning are the robustness and general applicability of the generated models, given the restricted sample size in any user study. We approach this issue by (randomly) splitting the original datasets into training and test sets (at the ratio 70:30), and only resampling the training set in each case. In order to do so, we use the resampling technique with replacement implemented in the machine learning toolkit WEKA [88]. This method of splitting and resampling allows us to avoid the problem of overfitting[10].

Overall, we carry out the above-mentioned method over ten runs for each class (i.e., each social circle and each tie strength category), each time randomizing the original dataset before splitting it into training and test sets. In each run, we generate a classifier based on the training set and analyze its performance on the corresponding test set. Finally, we average the results over all ten runs to obtain the classification results.

10 Overfitting implies that the generated model is overly tailored to the provided training data. This impacts the performance of the model negatively, causing the model to lose its ability to generalize to a wider range of datasets.

4.3.1.2 *Choosing the Machine Learning Algorithm*

For the machine learning algorithm, we consider *Support Vector Machines* (SVM), rule-based decision tree models—C4.5 and Random Forest—as well as the probabilistic Näive Bayes approach [117]. We test them on the original dataset using a 10-fold cross validation technique[11]. We use the corresponding implementations in WEKA (*LibSVM*, *J48*, *RandomForest*, and *NaiveBayes*, respectively) to test the above algorithms. Section A.1 in the appendix presents the empirical results obtained from the comparison of the above machine learning algorithms with respect to our dataset. Based on the obtained results, we observe that the Random Forest algorithm performs best with our dataset, achieving an average accuracy improvement of 5.71% over the other algorithms. This mainly attributes to the diverse nature of our dataset—with a considerable number of outliers—as well as the cluster-based approach employed by this algorithm. Therefore, we employ the Random Forest algorithm for the rest of our evaluation.

To better understand the results, we use the following evaluation metrics: accuracy, Cohen's Kappa κ, and the F1-score for true and false class estimation. The accuracy gives the ratio of the number of correctly estimated instances to the total number of instances in the dataset. Cohen's Kappa $\kappa \in [0, 1]$ shows the extent to which the estimated model deviates from the observed data, not accounting for random agreements. We use the F1-score in terms of F1-positive and F1-negative, which basically dictate the ability of an estimation model to differentiate true class instances from the non-class instances, and vice versa, respectively. Each of these metrics have a positive correlation with the performance of the estimation models, i.e., the higher the value of these metrics, the better is the performance of the models. In addition to these, we also include an analysis of the information gain achieved by each feature used in each dataset (using WEKA's implementation of the Ranker algorithm [88]). This mainly allows us to interpret the correlation factors assigned to each feature by the estimation models, corresponding to Equation 1.

Note that, unless otherwise stated, we employ the above method to prepare and evaluate our dataset extracted in *Phase II*, as well.

4.3.1.3 *Social Circle Analysis*

The classification results for each social circle based on the generated models are presented in Table 4. We observe that the average classification accuracy for the social circle *friend* when using the *AllComm* dataset is moderate at 61.22%. This can be attributed to the diversified perception of the term *friend* across different communities, which in turn is reflected in their communication patterns. We notice that, while the average accuracy does not change significantly for the *OnlyC* and *OnlyM* datasets, the average F1-positive score decreases considerably by 6.5% for the *OnlyC* dataset.

11 The cross validation approach provides a measure of the ability of an algorithm to generalize across a given dataset. As per the 10-fold cross-validation approach, the given dataset is randomly partitioned into 10 equal subsets; 9 of them are used for training the model using the given machine learning algorithm, and the remaining one fold is used as a test set for analyzing the performance of the trained model. This process is repeated 10 times with each of the 10 folds acting as the test set once.

Table 4: Evaluation Results for the Accuracy Analysis w.r.t. the Different Social Circles (*Phase I*)

Social Circle	Metric	AllComm	OnlyC	OnlyM
Friend	*Accuracy (%)*	61.22	61.08	62.84
	κ	0.2307	0.214	0.2489
	F1-pos. (%)	59.01	52.53	57.90
	F1-neg. (%)	62.85	66.83	66.40
Family	*Accuracy (%)*	81.49	82.97	65.68
	κ	0.3780	0.444	0.1216
	F1-pos. (%)	48.33	54.26	30.43
	F1-neg. (%)	88.69	89.39	77.03
Work	*Accuracy (%)*	90.68	90.95	69.32
	κ	0.2806	0.332	0.0632
	F1-pos. (%)	32.67	37.75	15.08
	F1-neg. (%)	94.94	95.11	81.23
Hobby	*Accuracy (%)*	76.08	70.68	68.92
	κ	0.4413	0.421	0.2674
	F1-pos. (%)	61.45	63.45	48.48
	F1-neg. (%)	82.45	75.17	77.59

This can be accredited to the relatively distinctive messaging pattern exhibited by *friend* instances. This is also reflected in the information gain analysis presented in Table 5, where we notice that the number of IM messages and the total number of communication events are the best positive indicators of *friend* instances.

While we achieve a significantly high average accuracy for the classification of both *family* and *work* instances at 81.49% and 90.68%, respectively, we notice that the κ and F1-positive values are relatively low at under 50% (whereas the F1-negative score is markedly higher). This is mainly due to the low number of representative instances for each of these classes in the overall dataset. However, we notice that the performance of the *OnlyC* dataset is distinctly better than that of the *OnlyM* dataset for both classes, with an accuracy increase of 17% and 21%, respectively. This can be attributed to the distinctive call-based interaction patterns in each of these classes, as well as a lack of distinctive interaction patterns in the messaging channels. We can also observe this pattern in the information gain analysis (see Table 5), where call-based features are the top three features in both classes. Furthermore, we also observe typical temporal patterns in the communication behavior of both *family* and *work* social circles, where *family* instances prefer communicating on weekends (here, Saturdays), and *work* instances prefer weekdays.

For the *hobby* instances, we obtain the best classification result at an average accuracy of 76.08% ($\kappa = 0.44$) for the *AllComm* dataset, which is also reflected in the high

Table 5: Information Gain Analysis w.r.t. the Different Social Circles (*Phase I*); Top Four Features based on Ranker [88]

Social Circle	Communication Feature	Info Gain (Corr. Valency)
Friend	No. of IM messages	0.095 (+)
	No. of all comm. events	0.08 (+)
	No. of all comm. events [14 days]	0.077 (+)
	No. of IMs [14 days]	0.077 (+)
Family	% Calls on Saturday	0.209 (+)
	% Call duration on Saturday	0.179 (+)
	Total duration of calls	0.172 (+)
	% Calls to overall comm.	0.152 (+)
Work	% Call duration on weekdays	0.319 (+)
	% Calls to overall weekday comm.	0.311 (+)
	% Calls on weekdays	0.311 (+)
	No. of IM messages	0.3 (-)
Hobby	% Calls to overall comm.	0.184 (-)
	No. of calls	0.184 (-)
	Total duration of calls	0.18 (-)
	% IMs read within 3 mins [14 days]	0.15 (+)

average F1-positive and negative scores (61.45% and 82.45%, respectively). However, we observe that, while the F1-positive score remains relatively constant, the average accuracy drops considerably (by 5.4%) for the *OnlyC* dataset in comparison to the *AllComm* dataset. This attributes to the distinctive call-based interaction patterns among *hobby* instances, as seen in the negatively correlated call-based features in the information gain analysis in Table 5. We can also discern that, while the average accuracy for both *OnlyC* and *OnlyM* datasets are similar, the F1-positive score drops significantly for the *OnlyM* dataset by 12.97% with respect to the *AllComm* dataset. This can be attributed to similar messaging patterns with *friend* instances. Interestingly, we see that people tend to respond to IM messages by *hobby* instances quite quickly, given that the percentage of IMs read within three minutes is a considerable contributing factor.

4.3.1.4 *Tie Strength Analysis*

The evaluation results with respect to tie strength are presented in Table 6, which are based on the Random Forest algorithm. We can discern that the best models are created when we use all communication features (*AllComm*), given the higher F1-positive score of 58.61% and 48.71% in comparison to *OnlyC* and *OnlyM*, respectively, for each tie strength dataset. This is mainly attributed to the higher influence of combinatorial features, such as the number of days of communication and the days

Table 6: Evaluation Results for the Accuracy Analysis w.r.t. Tie Strength (*Phase I*)

Tie Strength Category	Metric	AllComm	OnlyC	OnlyM
Strong	*Accuracy (%)*	75.81	75.95	74.59
	κ	0.4188	0.3715	0.2338
	F1-pos. (%)	58.61	53.05	38.23
	F1-neg. (%)	82.85	83.75	83.89
Very Strong	*Accuracy (%)*	80.13	81.60	71.47
	κ	0.3680	0.3338	0.2323
	F1-pos. (%)	48.71	44.15	39.51
	F1-neg. (%)	87.63	88.89	80.91

Table 7: Information Gain Analysis w.r.t. Tie Strength (*Phase I*); Top Four Features based on Ranker [88]

Tie Strength Category	Communication Feature	Info Gain (Corr. Valency)
Strong	No. of days of comm.	0.146 (+)
	Days since last comm.	0.129 (-)
	Total duration of calls	0.117 (+)
	No. of calls	0.116 (+)
Very Strong	No. of channels used	0.189 (+)
	Avg. no. of emoticons per IM [14 days]	0.184 (+)
	No. of days of comm.	0.169 (+)
	% Calls, SMS, and IMs on Saturday	0.169 (+)

since last communication, as seen in the information gain analysis with respect to the tie strength datasets in Table 7.

With respect to the *StrongWeak* dataset, we observe that, while the F1-negative score remains relatively constant for all three feature datasets, the F1-positive score drops significantly (by 20.38%) for the *OnlyM* dataset compared to *AllComm*. This seems to indicate that call-based features (such as total duration of calls, as seen in Table 7) have a higher influence on estimating *strong* instances than message-related features. This conforms with the results of existing work in the field of tie strength analysis where call intensity is considered a proxy for tie strength [156, 227]. Interestingly, we can see that this notion changes for the *VerystrongWeak* dataset, where the average number of emoticons used per IM message has a high positive correlation with the *very strong* instances. Another important feature in this regard is the number of communication channels used, which has the highest influence in differentiating between *very strong* from *weak* instances.

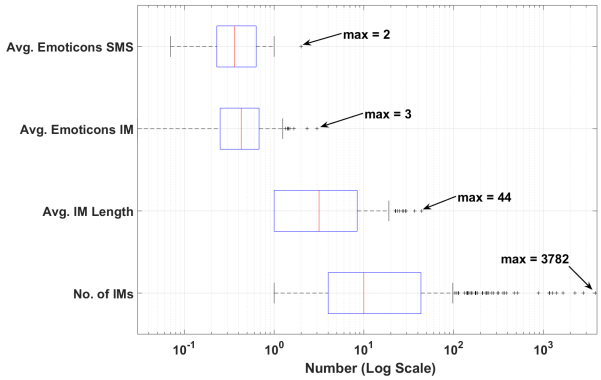


Figure 9: Distribution of the Main Extracted Features per Participant (*Phase II*)

4.3.2 *Results of* Phase II*: Analyzing Messaging Content*

As part of *Phase II*, we undertake an analysis of additional characteristics of instant messaging services and their influence in estimating the type and strength of a human relationship. In this study, we decided to exclude call-based features entirely, considering that our results from *Phase I* proved their applicability for estimating human relationships. Instead, we now focus on the additional messaging features concerning the usage of different emoticons as well as the feelings or sentiments behind the message contents.

We had a total of 26 participants who took part in the user study in *Phase II*. Considering the longer time span of this study as well as the fact that the monetary incentive motivated application usage for at least 4 weeks, we did not have to eliminate instances as in *Phase I*. Overall, we obtained a total of 547 instances, where each user assessed an average of 21 contacts (min = 3, max = 51, σ = 14.14). Based on the selected communication language in the questionnaires, 392 contacts communicate in German, 84 communicate in English, and the remaining 71 contacts usually communicate in a different language, giving us an overview of the demographic distribution of the participants.

Table 8 shows the dataset extracted in *Phase II*, indicating again that the number of *friend* instances is much higher than that of the other social circles. *Family* and *work* instances make up a higher percentage—18% and 14.5%, respectively—of the overall dataset compared to the dataset in *Phase I*. The number of instances for the social circle *significant other* is considerably higher, given the larger number of participants. In the social circle *others*, we obtained a varied set of contacts, including flatmates, university classmates, Internet acquaintances, doctors, and restaurants. In accordance with the study in *Phase I*, we exclude these two social circles from the following discussion. With respect to tie strength, we employ the same thresholds as in *Phase I* to obtain the two nominal datasets *StrongWeak* (threshold at 80%) and *VerystrongWeak* (threshold at 90%). In doing so, we obtain a total of 149 *strong* instances in the dataset *StrongWeak*, and 94 *very strong* instances in the dataset *VerystrongWeak* (see Table 8).

Table 8: The Extracted Dataset in *Phase II*

All Instances	Social Circles						Tie Strength (StrongWeak)		Tie Strength (VerystrongWeak)	
	Friend	Family	Work	S.O.	Hobby	Others	Strong	Weak	VeryStrong	Weak
547	285	99	82	18	81	37	149	398	94	453

Table 9: Channel Usage Based on Social Circle (*Phase II*)

		All	Friend	Family	Work	Hobby
Original Set		547	285	99	82	81
Instances with IM	⇒	353	214	50	48	56
	Linear decrease	353	183.92	63.89	52.92	52.27
	% Rel. change	–	+16.35	-21.74	-9.29	+7.13

Table 10: Evaluation Results for the Accuracy Analysis w.r.t. the Different Social Circles (*Phase II*)

Metric	Social Circle			
	Friend	**Family**	**Work**	**Hobby**
Accuracy (%)	59.33	76.34	80.55	78.84
κ	0.192	0.115	0.059	0.076
F1-pos. (%)	58.51	24.89	16.03	18.78
F1-neg. (%)	59.96	85.82	88.95	87.79

Our *Phase II* dataset consists of 33618 communication events, of which 31570 are IM messages and 2048 are SMS messages. Figure 9 shows the average distribution of messages per participant over both these channels. An interesting observation is the average number of messages containing emoticons in each of these messaging channels—the number of IM messages containing emoticons/emojis is 8 times higher than that for SMS messages. This can be attributed to the general proclivity towards IM (especially WhatsApp) and its emoticon set. Looking at the usage of the IM channel among the different social circles in Table 9, we can observe similar patterns as seen in the dataset extracted in *Phase I* (see Section 4.3.1). *Friend* and *hobby* instances tend to use IM more often (+16.35% and +7.13%, respectively), whereas *family* and *work* instances seem to exhibit certain reluctance towards the same (-21.74% and -9.29%, respectively).

For the analysis of the extracted dataset, we consider both the IM-based and SMS-based features as one set of messaging features, coming to a total of 62 features (compared to the 16 messaging features in *Phase I*). Just as we did in *Phase I*, we employ random resampling to balance out the imbalanced training sets, especially given the high number of *friend* instances compared to the other social circles, as well as the high number of *weak* instances compared to the *strong* and *very strong* instances. We split the original dataset into training and test sets at the ratio of 70:30, and apply the generated training set model on the corresponding test set for the respective analyses. Given the diverse and outlier-ridden nature of the dataset, just like in *Phase I*, we use the Random Forest algorithm in WEKA for supervised machine learning to generate the estimation models. Even here, we execute ten runs for each social circle/tie strength category, where we randomize the original dataset in each run. We use the same evaluation metrics used in *Phase I*—accuracy, Cohen's Kappa, F1-positive, and F1-negative—so that we can draw parallels between the two datasets. In the following, we describe the analysis of the new features based on messaging content in *Phase II* with respect to the social circles and tie strength categories.

4.3.2.1 *Social Circle Analysis*

Table 10 shows the evaluation results for the estimation of the social circles based on the Random Forest algorithm. We can notice that the correct estimation of the

Table 11: Information Gain Analysis w.r.t. the Different Social Circles (*Phase II*); Top Four Features based on Ranker [88]

Social Circle	Communication Feature	Info Gain (Corr. Valency)
Friend	No. of IM messages	0.039 (+)
	IM avg. message length	0.039 (+)
	Avg. general emoticons per IM	0.037 (+)
	% IM read within 10 mins	0.033 (+)
Family	No. of IM messages	0.105 (-)
	IM avg. message length	0.103 (-)
	IM avg. sentiment score [DE]	0.088 (-)
	% IM read within 10 mins	0.081 (-)
Work	No. of IM messages	0.108 (-)
	No. of IM messages [14 days]	0.072 (-)
	Outgoing SMS avg. sentiment score [DE]	0.047 (+)
	IM avg. message length	0.038 (-)
Hobby	No. of IM messages	0.092 (+)
	IM avg. message length	0.085 (-)
	IM avg. message length [14 days]	0.066 (-)
	No. of IM messages [14 days]	0.059 (+)

friend instances is still considerably difficult with an average accuracy of 59.33% (κ = 0.19). This is mostly caused by the diversity present within the social circle *friend*, as mentioned earlier in Section 4.3.1.3. The similar values of the F1-positive and negative scores are indicative of this diversity within this social circle. With respect to the information gain analysis (see Table 11), we can see that the overall number of IM messages and the average message length have the highest influence, albeit the amount of influence is considerably low (0.038). We also notice that the average number of general emoticons has a positive influence towards estimating *friend* instances.

Coming to *family* and *work* instances, we observe a similar issue faced in the analysis of the dataset in *Phase I*, where the messaging features do not possess any distinctive characteristics that allow the machine learning algorithm to precisely classify these instances from the others. While the average accuracy shows a significant increase compared to that in *Phase I*, the lack of uniqueness can be noticed in the lower values for Kappa (0.115 and 0.059, respectively) and F1-positive (24.89% and 16.03%, respectively) for both these social circles. This can also be observed in the information gain analysis, where most of the features have a negative influence on estimating these social circles. An interesting observation is that *work* instances seem to make use of the SMS channel for positive messages, at least in the German-speaking

Table 12: Evaluation Results for the Accuracy Analysis w.r.t. Tie Strength (*Phase II*)

Metric	Tie Strength Category	
	Strong	**Very Strong**
Accuracy (%)	72.56	81.65
κ	0.191	0.249
F1-pos. (%)	35.09	35.17
F1-neg. (%)	82.57	89.25

community, given the positive influence of the sentiment score for outgoing SMS messages.

The estimation models for *hobby* instances show a significantly improved average accuracy at 78.84% (compared to 68.92% in *Phase I*). However, the κ and F1-positive values are significantly poorer at 0.076 and 18.78%, respectively, indicating again the lack of unique messaging patterns in the *hobby* social circle compared to the others. The information gain analysis for *hobby* instances, however, shows familiar communication characteristics—the number of IM messages has a positive influence, whereas the average IM message length is negatively correlated with such instances.

4.3.2.2 *Tie Strength Analysis*

Table 13: Information Gain Analysis w.r.t. Tie Strength (*Phase II*); Top Four Features based on Ranker [88]

Tie Strength Category	Communication Feature	Info Gain (Corr. Valency)
Strong	Avg. emoticons per IM	0.053 (+)
	% IMs containing emoticon	0.049 (-)
	Avg. negative emoticons per IM	0.029 (-)
	No. of IM messages	0.026(+)
Very Strong	IM avg. message length	0.109 (+)
	Avg. general emoticons per IM	0.086 (+)
	IM σ of sentiment score [EN]	0.059 (+)
	Outgoing SMS avg. message length	0.056 (-)

The analysis of the messaging features with respect to tie strength yielded similar results to those in *Phase I*, as seen in Table 12. For the *StrongWeak* dataset, we notice that the values for κ and F1-positive are marginally lower than those in *Phase I*. However, for the *VerystrongWeak* dataset, we observe that the accuracy has increased considerably (about 10%), while the κ and F1-positive values have remained relatively constant in comparison to those in *Phase I*. This can be attributed to the fine-grained content-based features considered in *Phase II*, which is also seen in the information

gain anaylsis, where the number of general emoticons and the sentiment score in IM messages have a positive influence in distinguishing *very strong* instances (see Table 13). For the *StrongWeak* dataset, we notice that the average number of emoticons in IM messages has a positive correlation with *strong* instances, whereas the average number of negative emoticons has a negative correlation.

4.4 DISCUSSION

Based on the obtained results, we can ascertain that the two communication channels—calls and instant messaging—have an influential role in estimating human relationships. The distinctive characteristics of *friend* and *hobby* social circles in the IM channel does help in distinguishing them from the *family* and *work* social circles. Overall, we obtain an average accuracy of 77.36% ($\kappa = 0.33$) in estimating social circles, and an average accuracy of 75.81% ($\kappa = 0.42$) for identifying *strong* and *weak* instances, when we use the essential communication features from both channels.

Using only features based on call-based interaction, we obtain an average accuracy of 76.42% ($\kappa = 0.35$) for social circle classification, and 75.95% ($\kappa = 0.37$) for tie strength classification. On the other hand, using all messaging features, we only obtain a lower average accuracy of 73.76% ($\kappa = 0.11$) across all social circles and 72.56% ($\kappa = 0.19$) for tie strength, which is mainly caused by the lack of sufficient diversity in the communication behavior within each of the different categories. However, based on the information gain analysis for both social circle and tie strength categories, we notice that each category does exhibit a relatively significant behavior in each communication channel, which allows us to distinguish between them.

Looking at the complete picture, our results only depict the smartphone usage patterns of a select set of users in Germany. Our approach is based on the extraction of a sparse set of communication data to understand the characteristics of a given relationship. While the user study in *Phase II* was conducted over a sufficiently longer period of time than in the study in *Phase I*, a complete evaluation of human relationships requires a continuous collection of data over a period of years.

Considering that even meta-data extracted from user communication (i.e., the features) can contain sensitive information about the user, we employed two privacy-aware settings in our approach during the collection and analysis of the extracted data. Firstly, during our field studies, we used a unique hash-based[12] pseudonym for each user based on their Android-ID, and used this pseudonym while storing the extracted data on an external server. The extracted dataset allowed us to draw fruitful inferences on the estimation of user relationships from communication data, without allowing us any insight into the actual identity of the users behind the dataset. Secondly, during *Phase II*, apart from the collection of message-based features, we used some of the revelations gained in *Phase I* to provide the users with some initial estimations of their relationship with some of their smartphone contacts. In doing so, we did not send their data to any external server, thus performing a phone-local analysis.

12 https://en.wikipedia.org/wiki/SHA-1

Human relationships are in nature complex and dynamic structures. The social circle *friend* comprises a large variety of relationship types, including childhood best friends, schoolmates, and acquaintances at a party. Each of these sub-circles entail different communication behavior, on different communication channels. Ex-relationships, relationships over different time zones, as well as relationships with differing levels of engagement (e.g., project members, temporary roommates), to name a few, further complicate the creation of such estimation models.

Our research only focuses on a selection of IM application services—WhatsApp and Threema. There are many other IM applications available and used frequently nowadays, such as Facebook Messenger, Skype, and Google Hangouts. Furthermore, as mentioned earlier, we could only analyze the incoming messages in the IM applications due to the lack of the necessary API to access the application-internal data. Additional measures—e.g., understanding the touch patterns of smartphone users in messaging applications [198]—have to be taken to fully understand the implications of IM-based interactions on the estimation of user relationships.

Overall, our work focuses on the estimation of user relationships as the basis for understanding user sharing patterns and therefore, their privacy constraints. We can leverage findings in related literature to model the conditions in which people share information and to understand how the social circle and tie strength influence the data shared between people [51, 114, 153]. Furthermore, while social circle and tie strength have been proven to have a direct correlation with the sharing patterns within a relationship, one must also consider other factors while modeling one such system—surrounding conditions, user context, and the nature of the users themselves. Extroverts and introverts interact differently, as shown by de Montjoye et al. [143]. People also interact differently in groups, showing different communication and interaction patterns [107]. Adapting relationship estimation to dynamic user behavior is still an open research question.

In this thesis, we use the findings from our analysis of user relationships to generate the trust relations between the users, and therefore, between the devices in a D2D-based network. The trust relations provide us with a measure to quantify the amount of information shared between users, in accordance with the social circle and tie strength of the relationship at hand. We model the *dyadic* trust between two users based on their chief behavioral characteristics on the different communication channels under consideration—here, phone call and IM channels—using the concepts of *behavioral trust* [5, 190, 212]. In doing so, we focus on using the generated trust relations in realizing a privacy-aware environment for distributed CEP in the IoT, as discussed in the next chapter.

5 TRUST-BASED EXECUTION OF DISTRIBUTED CEP SYSTEMS

Our main revelation in Chapter 4 is the influential nature of distinct communication patterns on the estimation of different types of human relationships. In this chapter, we employ the key communication characteristics from Chapter 4 to derive the trust relations between the users, and thereby, between the devices themselves. These trust relations can be classified in a category of trust called *behavioral trust* [5, 190], where the trust between users is quantified based on certain characteristics of their communication or interaction behavior.

In general, D2D-based networks are built around the concept of physical proximity communication, so that the devices may exchange information directly among each other without having to rely on a cellular connection. This requires the knowledge of the device mobility patterns as well as the social relations between the devices and their owners to fully take advantage of the D2D environment [123, 154]. In turn, it is necessary to make the D2D-based network *socially-aware*, so as to adapt the network to the differing privacy constraints of the users involved. We use the concept of behavioral trust to establish a privacy-aware platform for distributed CEP in a D2D environment.

To this end, we introduce a privacy-aware placement algorithm for distributed CEP that accounts for the (behavioral) trust level between users. We control event dissemination and the placement of CEP operators with the help of a robust trust management model, that also accounts for trust evolution and prevents attacks on user privacy. Finally, we evaluate our model analytically as well as empirically as part of a D2D-based distributed system using smartphones.

5.1 BUILDING THE RIGHT CANDIDATE LISTS

Before we proceed with our trust-based approach for distributed CEP, we must first understand the main issues to be tackled in one such privacy-aware system. The main system prerequisite for the execution of an operator graph (see Section 2.1) in a distributed environment is the placement of CEP operators on available devices, such that the application requirements are fulfilled. These requirements can vary based on the application at hand—low latency, load balancing, low energy consumption, privacy awareness, etc. [121, 157, 187, 201, 231].

The issue of CEP operator placement basically translates to the building of the right candidate lists on each device. For example, for a low-latency application, the candidate list on each device includes other devices in the neighborhood along with the communication latency between them. Thus, when the CEP operators need to be placed on the devices, each device can choose a device in their candidate list that offers the lowest communication latency at the current point in time. The main objec-

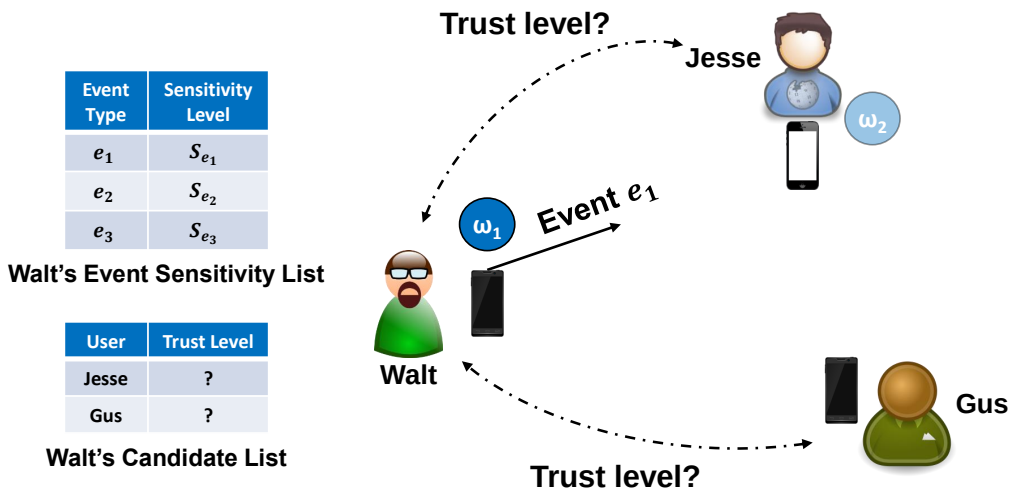


Figure 10: Illustration of the Problem Statement for Privacy-Aware Distributed CEP

tive of our work is to build these candidate lists on the basis of the trust level between the users who own the devices in the neighborhood, so as to enable a privacy-aware environment for distributed processing, as shown in Figure 10.

In general, CEP operates on historical atomic data (e.g., sensor data) and discovers higher-level patterns based on event observations. This requires the streaming of atomic events to processing devices, as well as the storage of these events on the devices for further processing. The privacy constraints of users depend on how sensitive each transmitted event is. Each event can have a different degree of sensitivity, or a *sensitivity level*. This again can vary from user to user, given that each user has a different perception of the sensitivity of events. For example, in Scenario A introduced in Section 3.1, Walt may consider his microphone data highly sensitive but may be willing to share his accelerometer information with other users without any concerns. The combination of events can further decrease or increase the sensitivity of the previous events. For example, an adversary may be able to infer Walt's stress level based on the combination of past microphone and location events [129, 184], but not by each event type individually. The dissemination of events must, therefore, performed in accordance with the event sensitivity level as well as the trust level of the possible event recipients.

In Section 1.1.1, we addressed the challenges faced in executing CEP in a D2D-based environment as a result of its dynamic nature and the presence of adversaries. The envisaged distributed CEP system must cope with variable data availability and privacy constraints in a possibly hostile environment, and yet allow for collaborative processing of context information among available devices. As mentioned in Section 2.2, trust characterizes the willingness of a person (trustor) to rely on the actions of another person (trustee), such that the trustor does not have any direct control over the actions of the trustee [168, 190]. In any hostile environment, it is necessary to account for the evaluation of trust as well as subsequent evolution of trust over

time. In effect, the users need to fill their candidate list with the correct trust values of the other users in the environment. In the following, we explain our proposed approach for the trust management model and detail its main properties that satisfy the above requirements.

5.2 TRUST-BASED DISTRIBUTED CEP USING TRUSTCEP

In our approach for a privacy-aware adaptation of distributed CEP, we develop a trust management model called TrustCEP based on behavioral trust, which accounts for the evaluation of direct trust between users as well as its evolution with time. As mentioned above, the trust level between users as well as the sensitivity level of the events disseminated dictate the placement of the CEP operators on the available devices.

Recall that an operator $\omega \in \Omega$—part of an operator graph G—processes the incoming events I_ω based on the prevailing operator rule model, and produces a set of outgoing events O_ω (see Sections 2.1 and 3.2.1). We consider that each event $e \in O_\omega \subset E$ has a certain sensitivity level S_e, which basically indicates how sensitive or private the event is to the given user. We assume that the users set a sensitivity level for each outgoing event type by themselves based on their privacy preferences as well as the context query that is currently processed. For example, in Figure 10, let τ_{Jesse} be the trust level assigned by Walt to Jesse. If S_{e_1} is the sensitivity level of the event e_1 output by operator ω_1 running on Walt's device, then the next operator ω_2 can be placed on Jesse's device if and only if $\tau_{Jesse} \geqslant S_{e_1}$. This ensures that devices which receive events have a high enough trust level to store and process them.

As can be discerned from the above discussion, the crux of the problem lies in determining the trust level τ_{Jesse} that Walt has in Jesse. In general, the key aspect behind our approach is the estimation and handling of trust in a distributed environment. Therefore, in TrustCEP, we put forth the following three contributions towards privacy-aware distributed CEP:

C5.1 We build on the observations made based on the analysis of user communication behavior to measure the direct trust relations between users. To this end, we utilize the results from our analysis in Chapter4 of the interaction history between users on synchronous and asynchronous communication channels, taking the key aspects of behavioral trust into consideration (see Section 5.2.1);

C5.2 For the evolution of trust, as part of the *Trust Recommendation Analyzer* (see Section 3.3), we develop an approach for robust trust recommendation where the users in the environment may recommend their trust values to others. Further, we also demonstrate how our approach is robust against collusion and on-off attacks (see Section 5.2.2). In summary, we present (i) how recommendation messages are exchanged between users, (ii) how we evaluate the credibility of the recommended trust values using a similarity-based approach, and (iii) how our approach prevents attacks on privacy;

C5.3 Finally, we detail a privacy-aware placement algorithm that makes use of the locally generated trust-based candidate list to find the right devices for the placement and execution of CEP operators (see Section 5.2.3).

5.2.1 *Establishing Direct Trust*

Researchers in the field of sociology have long studied the characteristics of trust and trustworthiness, and their correlation with the relationship between people [89, 207]. Furthermore, there has been a new wave of approaches that consider and embrace social awareness in the design of D2D-based networks towards 5G cellular systems [123, 141, 154, 228]. These approaches focus on the extraction of valuable information from social networks, including relationship details and recent communication history to estimate the *objective trustworthiness* of users, as mentioned in Section 2.2. The main goal is to incorporate the social relationships and trust level among devices to enhance the reliability and efficiency of IoT-based services and applications.

Recall from Section 2.3 that the notion of *tie strength*, or the closeness within a given relationship, is dependent on the intensity and reciprocity of the interactions between people. Typically, as established in related literature [5, 227], the most common indicators of trust and strength of a relationship are the number and duration of calls between users. This concept was validated and expanded in the results obtained in Chapter 4, where we observed that the characteristics of synchronous (e.g., call-based) and asynchronous (e.g., messaging based) interactions have a significant influence on the estimation of the relationship at hand.

To this effect, we propose an approach to establish the initial direct trust based on the aspect of behavioral trust between two users, in accordance with their historical interactions on synchronous and asynchronous communication channels. Synchronous communication includes, as the name suggests, all interactions that take place instantly, in real time, between two people—e.g., face-to-face communication, phone calls. In asynchronous communication, the sender and receiver do not have to be engaged at the same time, leaving room for more contemplation and deliberation in the responses—e.g., SMS communication and instant messaging. In our approach, we mainly consider the essential contributing factors in relationships and incorporate these to calculate the trust between the users.

Taking interaction intensity as one of the factors, we denote s and a as the interaction intensity on synchronous and asynchronous channels, respectively. This can, for example, be the number of calls and/or number of messages exchanged between two users, which attributes to varying degrees of closeness and hence, trust. In addition to these, we also consider factors that describe distinct usage patterns on the channels, and have been proven to have a significant influence towards the type and strength of a relationship—total call duration *dur* and the number of emoticons used in messages *emo* (see Section 4.3). To account for reciprocity, we introduce a degree of reciprocity $\rho \in [0, 1]$, that denotes the extent to which both users in a relationship are involved in the communication flow between them. This in turn depends on the

outgoing (*out*) and incoming events (*in*) in the respective channels—e.g., messages sent/received; time spoken/listened. All of these factors have to be weighed in appropriately, based on a weighting factor $w \in [0, 1]$, to estimate the trust between users. For example, the weighting factor $w_{s,out}$ accounts for the number of synchronous outgoing communication events.

For two given users v_i and v_j in the system, the (initial) direct trust level $\tau_{v_j}^{v_i}$ between them can be measured as a function of the synchronous and asynchronous components of their interaction history, τ_s and τ_a, respectively, as follows (we have left out the user indexes from the equations below for the purpose of comprehensibility):

$$\begin{aligned} \tau_s &= \frac{\rho_s \cdot (w_{s,out} s_{out} + w_{dur,out} dur_{out}) + (1 - \rho_s) \cdot (w_{s,in} s_{in} + w_{dur,in} dur_{in})}{s_{out} + s_{in} + dur_{out} + dur_{in}} \\ \tau_a &= \frac{\rho_a \cdot (w_{a,out} a_{out} + w_{emo,out} emo_{out}) + (1 - \rho_a) \cdot (w_{a,in} a_{in} + w_{emo,in} emo_{in})}{a_{out} + a_{in} + emo_{out} + emo_{in}} \end{aligned} \tag{2}$$

Consequently, the direct trust value of v_i towards v_j is given by,

$$\tau_{v_j}^{v_i} = w_{sync}\ \tau_s + (1 - w_{sync})\ \tau_a \tag{3}$$

where w_{sync} is the weighting factor that balances the direct trust between the synchronous and asynchronous trust values. If we consider a set of n_Υ users, each user $v_i \in \Upsilon \quad \forall i \in (1, 2, \ldots, n_\Upsilon)$ maintains a trust vector $T_\Upsilon^{v_i}$. The trust vector comprises v_i's trust level towards the other users in the environment, such that $T_\Upsilon^{v_i} = \{\ldots, T_{v_j}^{v_i}, \ldots\} \forall v_j \in \Upsilon \ni j \neq i$. Trust is generally considered a multi-dimensional construct [168, 190], as discussed in Section 2.2, such that it has varying values depending on the context—e.g., location, time, co-location, etc. For the sake of simplicity, we only consider one dimension of trust in our work. However, the concepts introduced here as well as the trust-based approach for operator placement can be easily extended to multiple dimensions, by introducing additional vector columns to the overall trust matrix, T_Υ.

In the above equations, the weighting parameters—$w_{x,y}$, w_{sync}, and ρ—may be dependent on each other, considering that the communication patterns themselves are correlated to each other to a certain extent. We assume that the users manipulate the different variables by themselves depending on the their proclivity to trust people and their disposition to build connections to other users. Given that the weighting parameters lie in the range $[0, 1]$, the resulting trust values also lie in the same range. We define 0 as the value representing *complete distrust*; 1 represents *complete trust*; and, 0.5 represents *neutral trust*. The more the trust value goes towards one of the extremities of the trust vector, the higher the (un-)trustworthiness of the corresponding user. We consider neutral trust as the trust level between users that do not have any or very few prior interactions.

5.2.2 *Trust Recommendations*

Our implementation of behavioral trust using historical interaction patterns primarily captures the trust $\tau_{v_j}^{v_i}$ that user v_i has towards user v_j, which we in turn term the direct trust between the two users. However, this concept does not help in establishing trust between users that have hardly interacted with each other. Restricting the trust evaluation to only previous interactions limits the devices in the respective candidate lists available for the placement of CEP operators, and therefore, reduces the viability of collaborative processing in a distributed environment.

In order to improve the scope of the system, we leverage the concept of *trust propagation* established in related literature [10, 127, 190]. Basically, taking the illustration in Figure 10 as an example, trust propagation accounts for the amount of trust Walt bestows upon Gus (whom, let us assume, Walt does not know) based on his existing trust level towards Jesse, and Jesse's trust level in Gus. It must be noted here that propagation of trust does not imply *transitivity* of trust, although the converse is always true [49, 190, 233]. While the amount of trust derived from propagation trust is context-dependent (e.g., if Jesse trusts Gus in a specific context, the trust propagated to Walt is also only applicable in that context), the principles of our single-context approach can be applied in a multi-contextual scenario, as well.

We incorporate the concept of trust propagation by allowing for the evolution of trust with the help of trust recommendations from other users. Therefore, the trust level of user v_i towards v_j, who have had minimal to no interaction with each other, depends on the trust recommendations from other users, $\tau_{v_j}^{v_t} \; \forall \; v_t \in \Upsilon \; \ni t \neq i, j$. As mentioned earlier, the incorporation of trust recommendations allows for an expansion in the number of devices available to occupy the candidate lists, and therefore, improve the efficiency of the distributed system on the whole.

However, the inclusion of trust recommendations makes the system susceptible to adversary threats in the form of collusion attacks. Adversaries can employ ballot-stuffing—i.e., improving their trust levels in the eyes of the other users—and bad-mouthing—i.e., reducing the trust levels of certain users to gain an advantage over them, as mentioned in Section 3.2.2.2. It is necessary to incorporate the incoming trust recommendations appropriately, in order to allow for improved system scope and yet prevent attacks on privacy due to the presence of adversaries.

In our work, we propose a robust trust recommendation approach that allows the benign users to prevent privacy attacks by adversaries in the system. To this end, we facilitate a user to (i) estimate the credibility of incoming trust recommendations, (ii) revise their trust vector based on the credible recommendations, and (iii) detect malicious behavior on the part of other users, including behavioral changes, e.g., from benign to malicious. In the following, we explain our approach in further detail with respect to the aforementioned points.

5.2.2.1 *Analyzing Trust Recommendations*

In our work, for a given user v_i, the corresponding trust recommendation vector is defined by $Rec_{\Upsilon}^{v_i}$, such that $Rec_{\Upsilon}^{v_i} \neg\Box \; T_{\Upsilon}^{v_i}$ (i.e., vector $Rec_{\Upsilon}^{v_i}$ is not necessarily equal

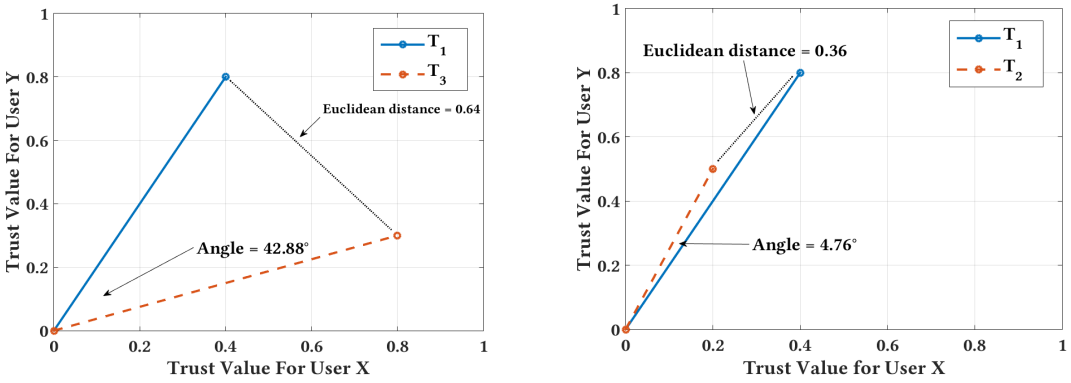
to vector $T_{\Upsilon}^{\upsilon_i}$). Each user can manipulate their recommendation vectors to suit their needs, depending on how they want to influence event dissemination in the system. We consider a *round-based* exchange of trust recommendation vectors among the users. The exact time interval for each round depends on the application at hand. For example, a financial application where trust plays a very important role may require trust transactions every minute; whereas, an application that involves social network analysis can have a larger interval for each round (e.g., 1 day). Each user υ_i modifies their trust recommendation vector to υ_j such that $rec_{\upsilon_t}^{\upsilon_i}|_{t=i,j} = \varnothing$, so that the mutual trust levels are not revealed to each other, considering that these are sensitive information.

Users benefit from trust recommendations such that they can update their trust level towards the other users, especially the ones who are (relatively) unknown. In order to incorporate them appropriately, we consider the *credibility* of incoming recommendations χ. We define the credibility of an incoming trust recommendation vector based on its *cosine-based similarity measure* with the other trust recommendation vectors—mainly to detect colluding users, as explained later in Section 5.2.2.2—as well as the trust vector of the recipient user to determine the revised trust level towards the other users.

Upon reception of trust recommendations, each user selects two sets of recommendation vectors for further analysis—(i) they first select the trust recommendations of the users whom they highly trust (e.g., $\tau > 0.9$), and (ii) further, they also consider those trust recommendation vectors among the remaining vectors that have a high credibility χ with respect to their own trust vector. In doing so, each user mainly considers those recommendation vectors that have a high degree of *semantic* similarity with its own trust vector [238]. Basically, we leverage on the notion that a user is more likely to follow other users who think alike, i.e., have similar trust vectors. For a user υ_i with trust vector $T_{\Upsilon}^{\upsilon_i}$, who receives the recommendation vector of user υ_j, $Rec_{\Upsilon}^{\upsilon_j}$, the credibility level of υ_j with respect to the received recommendation vector, $\chi_{\upsilon_j}^{\upsilon_i} \in [0, 1]$, is given by the cosine of the angle between the two vectors in an n-dimensional Cartesian coordinate system, where n is the number of users in the system, as in Equation (4).

$$\begin{aligned} \chi_{\upsilon_j}^{\upsilon_i} &= \frac{T_{\Upsilon}^{\upsilon_i} \cdot Rec_{\Upsilon}^{\upsilon_j}}{|T_{\Upsilon}^{\upsilon_i}| \cdot |Rec_{\Upsilon}^{\upsilon_j}|} \\ &= \frac{\sum_{t=1;t\neq i,j}^{n} \tau_{\upsilon_t}^{\upsilon_i} \cdot rec_{\upsilon_t}^{\upsilon_j}}{\sqrt{\sum_{t=1;t\neq i}^{n} (\tau_{\upsilon_t}^{\upsilon_i})^2} \cdot \sqrt{\sum_{t=1;t\neq j}^{n} (rec_{\upsilon_t}^{\upsilon_j})^2}} \end{aligned} \tag{4}$$

The dot product in the above equation does not apply for $t = i, j$ because of the exclusion of mutual trust values in the trust recommendations, as mentioned above, although it is inconsequential from a mathematical point of view. We consider a cosine-based approach to account for the relative predisposition that a user has towards the other users [221]. Basically, we compare the predisposition that a user has towards others and use this knowledge to select the trust recommendation vectors



(a) Starkly Dissimilar Trust Vectors

(b) Relatively Similar Trust Vectors

Figure 11: Comparison Between Cosine-based and Euclidean-based Similarity Measure

that are most credible with respect to the user. Different weighting factors for estimating the trust vectors, as in Equations (2) and (3), will lead to different trust values for each user.

Consider Figure 11, where we depict the comparison between two non-similar trust vectors (Figure 11a) and two similar trust vectors (Figure 11b), with respect to the cosine of the angle between them. We observe that, in Figure 11a, the trust vector T_3 is significantly dissimilar to trust vector T_1 because of the large angle between them (42.88°). In this case, even the Euclidean distance between the two vector tips will provide a reasonable estimate of the (dis-)similarity between the two vectors, given the large value of 0.64. However, in Figure 11b, the two trust vectors are very similar to each other, since they both exhibit a similar proclivity towards other users, given the low angle between them, $\cos^{-1}(\chi^{T_1}_{T_2}) = 4.76°$. In this case, the Euclidean distance between the two vectors provides a false estimate of the similarity between the vectors, given its disregard to the inclination of the trust vectors and a relatively large value of 0.36.

Each user considers the recommendation vectors that have a credibility level $\chi \geqslant \chi_{threshold}$. Among the selected recommendation vectors, each user measures the *divergence* δ (or the displacement) of the recommended trust values from those in their own trust vector, to identify the trust values that may need adjustment. The mean divergence of the recommended trust values is then passed through a weighting function based on the credibility level with respect to the corresponding recommender, which results in the *trust divergence* with respect to a given user. For example, if user v_i is measuring the trust divergence with respect to the existing trust value towards v_j, then the recommended trust values from the other users $rec^{v_t}_{v_j}|_{t\neq i,j}$ are considered, if they satisfy the credibility check mentioned above. The trust divergence for v_j with respect to v_i is given by Equation (5).

$$\delta^{v_i}_{v_j} = \frac{\sum_{t=1;t\neq i,j}^{n} \chi^{v_i}_{v_t} \cdot (\tau^{v_i}_{v_j} - rec^{v_t}_{v_j})}{\sum_{t=1;t\neq i,j}^{n} \chi^{v_i}_{v_t}} \tag{5}$$

We adopt an *additive increase, multiplicative decrease* approach to modify the trust values based on the trust divergence. This approach is based on the well-known principle used for TCP congestion control, which allows a system to get to equilibrium in a fair manner [104]. In our work, this offers a conservative notion for modifying the trust values, such that the (suspected) adversaries are punished heavily for any discrepancies. In this manner, the benign users are assisted in their trust evaluation of the other users, allowing them to hinder any attacks on privacy by the adversaries.

In order to avoid a ceaseless modification of the trust values, we consider a threshold value $\delta_{threshold}$, such that the trust values are modified only when the absolute value of trust divergence exceeds the above threshold. In other words, the trust value of v_i towards v_j, $\tau_{v_j}^{v_i}$, is only modified if $|\delta_{v_j}^{v_i}| \geqslant \delta_{threshold}$. We consider two coefficients to account for the additive increase, multiplicative decrease principle mentioned above—trust increase coefficient τ_{inc}, which attributes to the additive increase factor, and trust decrease coefficient τ_{dec}, which attributes to the multiplicative decrease factor. Effectively, the trust value $\tau_{v_j}^{v_i}$ is modified using Equation (6) as follows:

$$\tau_{v_j}^{v_i} = \begin{cases} \tau_{v_j}^{v_i} + \tau_{inc}, & \text{if } \delta_{v_j}^{v_i} < 0 \\ \tau_{v_j}^{v_i} / \tau_{dec}, & \text{otherwise} \end{cases} \ni |\delta_{v_j}^{v_i}| \geqslant \delta_{threshold} \tag{6}$$

As part of the analytical evaluations in Section 5.3.1, we assess the consequences of different conservative and liberal values for both τ_{inc} and τ_{dec}.

5.2.2.2 *Preventing Attacks on Privacy*

While the above sections show how the incoming trust recommendations should be analyzed in order to modify one's own trust vectors, this section focuses on mitigating adversary attacks by identifying the benign and malicious users in the system, and appropriately handling their trust values. Going by the adversary models described in Section 3.2.2.2, we now explain how our approach handles such adversary actions. The corresponding approach is also described in Algorithm 1.

Collusion Attacks. As mentioned earlier, adversaries attempt to influence the system functionality by colluding with other users, and adjusting their trust recommendations to suit their needs. By circulating falsified trust recommendations, they attempt to manipulate the trust vectors of benign users. To detect such adversary attacks, each user first measures the credibility level of each incoming recommendation vector against the other incoming recommendation vectors (Line 7). Based on the obtained values, each user v_i determines the *cumulative recommendation credibility measure*, $\chi_{v_i}^{v_j}$, for each user v_j. Each user then uses its collective set of cumulative recommendation credibility measures, χ_{Rec}, to detect the users that have provided the most uncorrelated values (Lines 9–13). We employ a standard deviation based outlier detection approach for identifying those users whose credibility level varies strongly from that of the rest of the recommending users (Line 13). The recommendation vectors of these users are rejected from further analysis. Further, in order to hinder any possible future attempts from these (suspected) adversaries, they are

Algorithm 1 : Preventing Privacy Attacks (w.r.t. User υ_i)

Description : Algorithm used to process incoming trust recommendations and detect adversaries based on the credibility level of the incoming recommendations

Variables : $T_\Upsilon^{\upsilon_i} \leftarrow$ current trust vector of υ_i
$markCount \leftarrow \varnothing$
$\chi_{Rec} \leftarrow \varnothing$ `// set of cumulative recommendation credibility measures`
$recHist \leftarrow \varnothing$ `// history of received recommendations`
$\chi_{threshold} \leftarrow$ credibility level threshold

```
1  for roundNumber = 1 to n_round do
2      for j = 1 to n_Υ and j≠i do
3          function receiveRecommendations(Rec_Υ^{υ_j}) ∋ rec_{υ_{i,j}}^{υ_j}=∅
4              if calcSim(Rec_Υ^{υ_j}, recHist[j]) < χ_threshold then
5                  if roundNumber > 1 then markCount[j] += 1;
6              else if markCount[j] < malValue then
7                  χ_{υ_j}^{υ_i} ← calcSim(T_Υ^{υ_i}, Rec_Υ^{υ_j});
8              recHist[j] ← Rec_Υ^{υ_j};
9      for j = 1 to n_Υ and j≠i do
10         for k = 1 to n_Υ and k≠i do
11             χ_Rec[j] += calcSim(Rec_Υ^{υ_j}, Rec_Υ^{υ_k});
12     for j = 1 to n_Υ and j≠i do
13         if |(μ_{χ_Rec} − χ_Rec[j])| > σ_{χ_Rec} then
14             markCount[j] += 1;
15     if roundNumber > n_wait then
16         for each j ∈ markCount < malValue do
               // loyalty-based trust
17             markCount[j] −= 1;
18             τ_{υ_j}^{υ_i} += τ_inc ∋ max(τ) = 1;
19             modifyTrustBasedOnDivergence(T_Υ^{υ_i}, χ);
20         for each j ∈ markCount > malValue do
21             τ_{υ_j}^{υ_i} /= τ_dec;
```

marked (Line 14). Evidently, it may be possible that certain benign users are marked unfairly. Therefore, each user waits a few rounds, n_{wait}, before taking action against the suspected adversaries.

The marked users are punished based on the number of marks they obtain due to their discrepancies in their credibility level. If the mark count goes beyond the *malicious threshold*, *malValue*, then the corresponding users are marked as malicious and their trust level is decreased based on the multiplicative decrease principle mentioned in Section 5.2.2.1 (Line 21). On the other hand, it is also necessary to vindicate

the benign users that have been marked as suspicious, if their mark count does not exceed the malicious threshold. To this end, we introduce the concept of *loyalty-based trust*, where the trust value of (presumably) benign users is increased by the trust increase coefficient τ_{inc}, if the waiting period n_{wait} has elapsed (Lines 17–18). The effective value for the malicious threshold *malValue* plays a very important role in deciding the functionality of the system—a larger value may allow the adversaries to remain undetected for a longer period of time; a shorter value may lead falsely categorizing benign users as adversaries. We evaluate the implications of *malValue* on the trust values in Section 5.3.1.

On-off Attacks. To account for on-off attacks, we assume that adversaries can behave benignly for a given period in time, n_{benign} before starting to collude with other users, as in the above case. One can handle such attacks by penalizing the adversaries strongly, using a high value for the trust decrease coefficient τ_{dec}, as in Equation (6). We also tackle frequent aberrations in the trust recommendations by measuring the credibility level among the past recommendation vectors sent by a given user (Line 4). Each user marks those users where $\chi < \chi_{threshold}$ and deals with them as described above. We shall prove the effectivity of the above approach in the analytical evaluation in Section 5.3.1. However, we do not account for all types of on-off attacks. For improved approaches to combat more complicated versions of on-off attacks, we refer the interested readers to the work by Chae et al. [43].

5.2.3 *Privacy-Aware Operator Placement and Execution*

In this section, we explain our approach to use TrustCEP for privacy-aware placement of CEP operator graphs in a distributed environment. Algorithm 2 describes the approach, primarily from the perspective of the *leader* v_i (see Section 3.2.2).

At the beginning, v_i, as well as the other users in the environment, build their respective candidate lists by searching for the neighboring users. It must be noted here that the candidate lists do not have to be restricted to users in the immediate neighborhood, but can include all users who have been encountered, in general. However, for the processing of a given query, only the users in the immediate vicinity are considered. Based on the historical interaction patterns with the corresponding users, v_i then sets up the trust values for the users in the candidate list, producing the trust vector $T_{\Upsilon}^{v_i}$. These trust values are then updated using the current trust recommendation vectors $Rec_{\Upsilon}^{v_j} \forall v_j \in \Upsilon \ni j \neq i$, as explained in Algorithm 1.

For a given context query Q, the *leader*, as well as the other users taking part in the said query, set up the sensitivity levels S_e for the events $e \in E$ to be disseminated. This is done in accordance with their respective privacy preferences. Based on the established sensitivity levels S as well as the query at hand, the leader then sets up the corresponding operator graph G (Lines 7–8), similar to related work on distributed CEP [54, 157].

If there are no users discovered in the vicinity (Line 1), then v_i executes the generated operator graph on their own device. As mentioned in Section 3.2.2, we mainly consider user smartphones as the sensing and processing devices in our system. If

Algorithm 2 : Trust-Based Operator Placement (v_i as Leader)

Description : Algorithm used by v_i as leader to incorporate the trust relations in the placement of CEP operators on the available devices

```
1  Υ ← findNeighboringUsers();
2  if Υ ≠ ∅ then
3      T_Υ^{v_i} ← evaluateDirectTrust(Υ);
4      Rec_Υ ← obtainTrustRecommendations(Υ);
5      T_Υ^{v_i} ← modifyTrustValues(T_Υ^{v_i}, Rec_Υ);
6  function rcvQuery(Q)
7      S ← setupEventSensitivity(Q);
8      G ← establishOperatorGraph(Q, S);
9      if Υ = ∅ then
10         executeOperatorGraph(G);
11     else
12         Γ_Υ^{v_i} ← establishPlacementGraph(G, T_Υ^{v_i}, Rec_Υ);
13         sendPlacementRequests(Γ_Υ^{v_i}, Υ);
14 function rcvResponse(Υ, C_Υ)
15     if C_Υ ≠ ∅ then // trust conflict
16         Γ_Υ^{v_i} ← modifyPlacementGraph(Γ_Υ^{v_i}, C_Υ);
17         sendPlacementRequests(Γ_Υ^{v_i}, Υ);
18     else
19         executeOperatorGraph(Γ_Υ^{v_i});
```

the candidate list of the leader v_i is not empty, then v_i sets up the placement graph $\Gamma_\Upsilon^{v_i}$ based on the established trust vector as well as the recommendation vectors received from the other users. The recommendation vectors are used to understand how the other users trust each other and determine the appropriate devices for collaboration. Upon setting up the placement graph, v_i assigns the operators to the corresponding user devices in the form of placement requests (Lines 12–13).

Since each user can have different sensitivity levels for their outgoing events, each of the neighboring user devices that receives a placement request first checks if their privacy constraints may be violated based on the current placement graph. If so, a conflict message C_Υ is sent back to the leader, to alert them about the said conflict and also inform them about any possible substitute user device that can manage the execution of the conflicted path (Line 15). The leader modifies the placement graph based on the trust conflict vectors sent by the other users, and then sends out the new placement requests. If there are no conflicts left and the trust negotiations have been concluded, the leader initiates the execution of the operator graph.

5.3 EVALUATION

We evaluate the designed trust management model TRUSTCEP with respect to two aspects: (i) We analytically analyze the efficacy and applicability of TRUSTCEP for enabling trust evaluation and evolution in a possibly hostile environment with colluding adversaries, and (ii) we evaluate the performance of TRUSTCEP in terms of battery consumption and network data usage as part of a prototypical implementation of a distributed CEP system, spanned over a smartphone-based D2D network for context processing. In the following, we shall describe our evaluation set-up and approach for both these aspects and present the corresponding results.

5.3.1 *Analytical Evaluation of* TRUSTCEP

We conduct the analytical evaluation of TRUSTCEP by undertaking a simulation-based approach, where we examine the (average) trust value obtained by the different users in the environment, by varying the behavior of the adversaries and their attacks.

5.3.1.1 *Analytical Evaluation: Approach Overview*

As explained in the previous sections, we mainly focus on the evaluation of TRUSTCEP against collusion and on-off attacks. In case of collusion attacks, we assume that there is a group of adversaries among the benign users, who collude among themselves to manipulate the trust values in the system. These adversaries either increase or decrease the trust values in their recommendation vectors, depending on whom they send their recommendation vectors. A benign user v_i will receive a recommendation vector $Rec_{\Upsilon}^{v_j}$ from an adversary v_j, where the trust values for other benign users are reduced by 0.5 (*bad-mouthing*), and that for the other adversaries is increased by 0.5 (*ballot-stuffing*), subject to $\tau \in [0, 1]$ in both cases.

In case of on-off attacks, we design the adversaries to pretend to be benign for N_{benign} rounds before they turn malicious and start to collude among themselves. However, in this case, instead of infiltrating every benign user in the environment with falsified trust recommendation vectors, we design the adversaries to only provide falsified data to a (randomly) selected set of benign users. In turn, we analyze the effects of partial infiltration on the part of the adversaries.

In both cases, the initial direct trust vectors for each user are generated at random using a uniform distribution function between 0.5 and 1, covering all possible trust values from trust neutrality to complete trust. We purposely choose this range based on the premise that the users do not usually have any distrusted neighbors at the beginning, without prior interactions. Each simulation setting is executed for a fixed number of rounds, by modifying the different variables introduced and discussed in Section 5.2. The key variables that play an important role in the ensuing evaluation are presented in Table 14, where the default values of the variables are written in bold and underlined, unless otherwise stated in the discussion below. In each sim-

Table 14: Default Parameters for the Analytical Evaluation of TrustCEP

Parameter	Value
Number of users, $n_{\mathcal{Y}}$	20
Number of simulated rounds, n_{round}	20
Waiting period, n_{wait}	0, **4**, 10
Benign period (for on-off attack), n_{benign}	10
Adversary population (%)	**40**, 70
Similarity threshold, $\chi_{threshold}$	0.95
Divergence threshold, $\delta_{threshold}$	0.1
Trust increase coefficient, τ_{inc}	0.025, **0.05**, 0.1
Trust decrease coefficient, τ_{dec}	0.5, **0.75**, 0.9
Malicious threshold, *malValue*	$100\tau_{inc}$

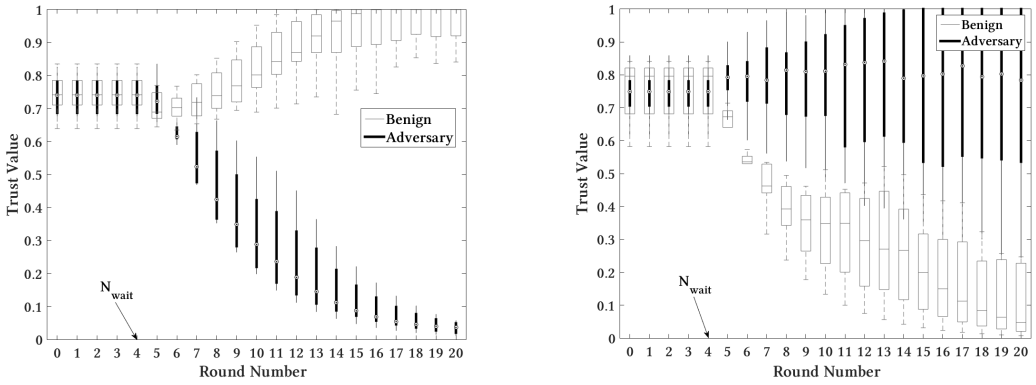
ulation run, we observe the average trust value assigned by the benign users to the adversaries as well as the other benign users.

5.3.1.2 TrustCEP *against Collusion Attacks*

We first analyze the performance of TrustCEP against collusion attacks with increase in the number of rounds executed, and observe the trust values obtained by the benign users and adversaries. Figure 12 shows the results obtained for two sample cases with different adversary populations in the form of box plots—the line in the middle of the box represents the median of the corresponding trust value distribution; the top and bottom edges of the box represent the 25^{th} and 75^{th} percentiles; and, the whiskers represent the 5^{th} and 95^{th} percentiles.

In Figure 12a, we see the trust value distribution for a sample case with 40% adversaries among the 20 users in the environment, representing a case with the adversaries in the minority. We observe clearly that the trust value obtained by the adversaries reduces exponentially with increase in the number of rounds, after the waiting period n_{wait} has elapsed. Over 75% of the adversaries receive a trust value of 0.5 or below by round 9. This is mainly attributed to the standard deviation based outlier detection (Line 13 in Algorithm 1) and the multiplicative decrease coefficient τ_{dec}, as also seen in the decline of the variance of the distribution over the number of rounds. We also observe that the trust value towards benign users improves steadily with increase in the number of rounds, after a minor drop after n_{wait}. This can be accredited to loyalty-based trust (Line 18 in Algorithm 1), with over 50% of the benign users achieving almost the complete trust level by round 16.

On the other hand, Figure 12b shows the trust value distribution for a sample case with 70% adversary population, such that the adversaries are in the majority. We observe an inverse outcome in the trust value distribution with increase in the



(a) Minority Adversary Population: 20 Users, 12 Benign, 8 Adversary

(b) Majority Adversary Population: 20 Users, 6 Benign, 14 Adversary

Figure 12: Trust Value Distribution upon Collusion Attack for Two Sample Cases with Different Adversary Populations

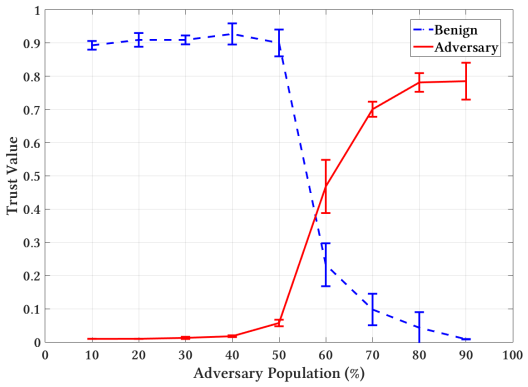


Figure 13: Trust Value Distribution upon Collusion Attack over Increasing Adversary Populations

number of rounds. After the initial waiting period of 4 rounds, we notice that the trust value obtained by the benign users decreases steadily, such that more than 50% of the benign users obtain an average trust value of 0.3 or below by round 12. This is mainly caused by the standard deviation based outlier detection approach, where now the benign users are suspected to be malicious and assumed to be providing falsified trust recommendation vectors. As seen in the trust value distribution among the adversaries, where almost 50% of the adversaries obtain an average trust value of 0.8 or higher at the end of 20 rounds. However, we notice that the variance in the obtained trust values among the adversaries is quite high, which can be attributed to the different initial trust values assigned to them by the benign users as well as the differing credibility levels, accordingly.

The results in Figure 12 only show the changes in trust value distribution across benign users and adversaries for two sample cases, with increase in the number of rounds. In Figure 13, we see the performance of TrustCEP in terms of the trust value distribution for benign users and adversaries for different adversary populations,

from 10% to 90%. In this plot, we present the average trust value obtained by the benign users and adversaries at the end of 20 rounds. For each adversary population, we randomized the initial trust value distribution 10 times and present the average and standard deviation values (the whiskers), accordingly.

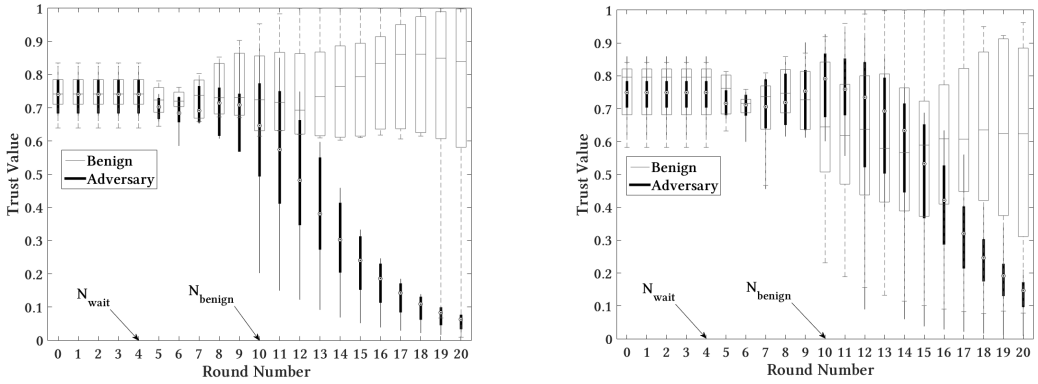
Overall, we observe a similar pattern as seen in the above sample plots for 40% and 70% adversary populations. For adversary populations of 50% or lesser, we notice that the benign users obtain a high trust value of 0.9 or higher, whereas the adversaries obtain a very low trust value of less than 0.1 on average. However, we observe that the situation changes when the adversaries become the majority population, especially when the adversary population increases beyond 70%. This behavior is an expected observation based on the information exchanged among the users in our trust management model. Interestingly, we observe that the final average trust value obtained by the adversaries remains at trust neutrality when the adversary population is at 60%, although the average trust value obtained by the benign users drops significantly compared to the case with 50% adversaries.

5.3.1.3 TrustCEP *against On-Off Attacks*

Similar to the analysis of TrustCEP against collusion attacks, in this section, we analyze the performance of TrustCEP against on-off attacks. In the first step, we considered the same sample cases that are used for the collusion attack analysis, where the adversary population is in the minority (40%) and in the majority (70%). We retained the same set of initial trust values for these two sample cases. Here, unlike the above analysis, the adversaries remain benign for $n_{benign} = 10$ rounds before turning malicious and colluding with fellow adversaries in the system. Furthermore, as mentioned earlier, in this case, the adversaries only send falsified trust recommendations to a (randomly) selected set of benign users, accounting for a partial infiltration of the environment. The results of the on-off attack analysis for the two sample cases are presented in Figure 14.

Figure 14a presents the trust value distribution over increasing number of rounds for all users, with 40% of them being adversaries. We observe that, similar to the results in the collusion attack analysis, the trust value obtained by the adversaries decreases rapidly, owing to the multiplicative decrease coefficient τ_{dec}. However, in this case, this rapid decrease of trust only occurs after n_{benign} rounds, when the adversaries switch their behavior from benign to malicious. We notice that the trust values obtained by the adversaries remains almost constant in the phase between n_{wait} and n_{benign}. Interestingly, we can also discern that the average trust value obtained by the benign users is affected considerably due to the brief falsehood on behalf of the adversaries, where around 50% of the benign users obtain an average trust value of 0.8 or lower, and certain benign users are also considered malicious at the end of 20 rounds.

The results of the on-off attack analysis for the sample case where the adversaries are in the majority are presented in Figure 14b. We observe that while the average trust value obtained by the adversaries increases steadily after n_{wait} rounds, it drops quite drastically after they switch their behavior to malicious after n_{benign} rounds.



(a) Minority Adversary Population: 20 Users, 12 Benign, 8 Adversary

(b) Majority Adversary Population: 20 Users, 6 Benign, 14 Adversary

Figure 14: Trust Value Distribution upon On-Off Attack for Two Sample Cases with Different Adversary Populations

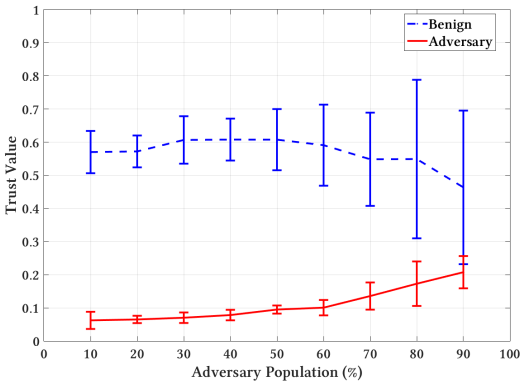


Figure 15: Trust Value Distribution upon On-Off Attack over Increasing Adversary Populations

More than 75% of the adversaries obtain an average trust value of 0.3 or lower by round 18. This is mainly attributed to the aforementioned property of selective infiltration on the part of the adversaries, such that the benign users can identify the adversaries based on the recommendation vectors received from the other users. We also observe that the average trust value obtained by the benign users falls in the trust neutrality range, with the median trust value lying at just above 0.6 after 20 rounds. This is mainly due to the previous benign behavior by the adversaries that led to certain benign users to be marked as malicious, having exceeded the malicious threshold *malValue*.

Figure 15 shows the trust value distribution for all users under an on-off attack over an increasing adversary population from 10% to 90%, similar to the collusion attack analysis presented in Figure 13. Similar to the results obtained in the above sample cases for minority and majority adversary populations, we observe that the adversary users obtain a significantly low average trust value over all adversary populations, primarily due to the consequences of the above-mentioned selective in-

filtration of the environment. When the adversaries are in the majority, their average trust value increases slightly, but it is nevertheless around 0.2, which is noticeably low. Among the benign users, we observe that the brief benign phase by the adversaries does affect their average trust values. The average trust value of the benign users remains around 0.6 for all adversary populations up to 60%. A further increase in the adversary population leads to a slight deterioration in the average trust value of the benign users, dipping below 0.5 after 20 rounds for the case with 90% adversaries. The relatively low average trust value of benign users (c.f. Figure 13) is again attributed to the standard deviation based outlier detection, which leads to certain benign users being falsely judged as malicious.

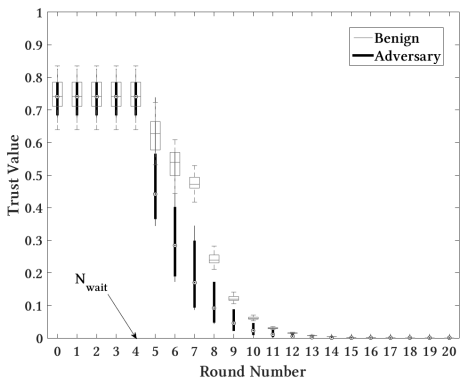
5.3.1.4 *Influence of Trust Modification Coefficients: Conservative Case*

In this section, we analyze the influence of the trust modification coefficients—the trust increase and decrease coefficients, τ_{inc} and τ_{dec}, respectively—on the trust value distribution under the influence of a collusion attack. In this section, we restrict our focus to the analysis of collusion attacks; for the results from the analysis of on-off attacks with respect to the trust modification coefficients, we request the interested readers to refer to Section A.2 in the appendix.

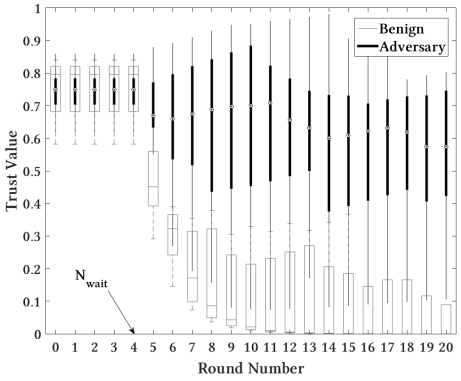
For the subsequent analysis, we consider two evaluation settings—(i) a *conservative* setting with lower values than before for the trust modification coefficients; and (ii) a *liberal* setting with considerably higher values than before. For the *conservative* case, we set the (additive) increase coefficient τ_{inc} to 0.025 and the (multiplicative) decrease coefficient τ_{dec} to 0.5. We proceed in the same way as in Section 5.3.1.2, such that we consider the same sample cases along with the same set of initial trust values.

Figure 16 shows the results for the trust value distribution in the conservative case using box plots, as before. In Figure 16a, where the adversaries are in the minority at 40%, we observe that the average trust value obtained by the adversaries decreases even more rapidly than in the non-conservative case, with over 75% of the adversaries obtaining an average trust value of 0.3 or lower by round 7, compared to round 13 in Figure 12a. However, we also observe that the average trust value obtained by the benign users reduces drastically with increase in the number of rounds, with all benign users obtaining an average trust value of 0.3 or lower by round 8. This is a natural consequence of using conservative settings for the trust modification coefficients. Another contributing factor is the resulting low value for the malicious threshold, *malValue* which is set to $100\tau_{inc} = 2.5$. The low value for the malicious threshold leaves little room for any aberrations in the trust recommendations from the benign users.

When the adversaries are in the majority at 70%, as seen in Figure 16b, we observe that the average trust value of the benign users reaches a lower value after 20 rounds than in the non-conservative case, with all benign users obtaining a trust value of 0.1 or lower. However, given the conservative approach towards increasing trust values, we observe that the average trust value obtained by the adversaries settles down around the trust neutrality value of 0.5. The large variance in the obtained trust



(a) Minority Adversary Population: 20 Users, 12 Benign, 8 Adversary



(b) Majority Adversary Population: 20 Users, 6 Benign, 14 Adversary

Figure 16: Trust Value Distribution upon Collusion Attack for Conservative Trust Modification Coefficients

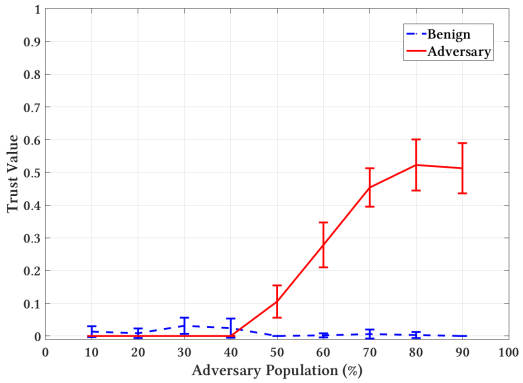


Figure 17: Trust Value Distribution upon Collusion Attack over Increasing Adversary Populations for Conservative Trust Modification Coefficients

values among the adversaries can again be attributed to the different initial trust values and the differing credibility levels, as in the non-conservative case.

The trust value distribution for the conservative case across all adversary populations in Figure 17 shows us that the benign users suffer most, obtaining a very low average trust value throughout the entire simulation. This is mainly due to the low value of the malicious threshold, which in turn is breached in all adversary populations because of two main reasons—the lower credibility levels of the benign users against each other, and the falsified trust recommendations sent by the adversaries that are assumed to be true. We notice that the average trust value received by the adversaries increases steadily from the point when the adversaries go into the majority. It finally settles down at the trust neutrality value of 0.5, which concurs with our observations in the majority sample case above (see Figure 16b).

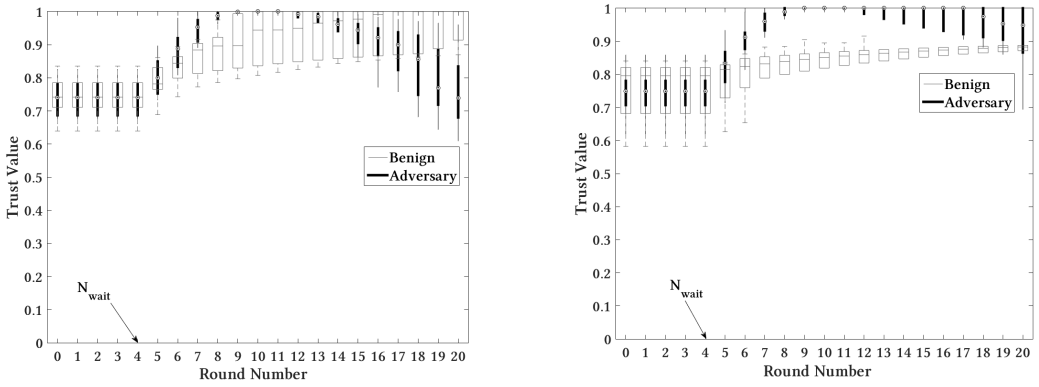
5.3.1.5 *Influence of Trust Modification Coefficients: Liberal Case*

For the *liberal* case, we set the trust (additive) increase coefficient τ_{inc} to 0.1 and the (multiplicative) decrease coefficient τ_{dec} = 0.9, such that they are considerably higher than those used for the normal analysis (see Table 14). Figures 18 and 19 present the results of the corresponding analysis of the modification coefficients, using the same approach as before (i.e., same initial trust values and same sample cases).

In Figure 18a, we observe that the benign users in the system have a steady increase in the average trust value they obtain, mainly accounting for the loyalty-based trust, as seen in Section 5.3.1.2 (c.f. Figure 12a). Given the liberal settings in this case, we observe that all benign users obtain an average trust value of at least 0.8 by round 9, unlike the non-liberal case, where it takes 17 rounds. Interestingly, we observe that the adversaries obtain a very high average trust value at the beginning until round 11, but then slowly lose their trust value. We notice that there is a steady decline in their average trust value from round 12 onward. This can be attributed to the high trust the benign users have bestowed upon the other benign users by this round, leading to the adversaries being discovered based on the standard deviation based outlier detection approach.

We see the results of the analysis of the liberal case for a majority adversary population in Figure 18b. Quite as expected, we notice that the average trust value obtained by the adversaries increases steeply within the first 5 rounds after n_{wait}, causing the adversaries to obtain complete trust level by round 9. However, we observe that even the benign users obtain increasing trust values with increase in the number of rounds, albeit at a relatively lower rate than that in case of the adversaries. This can be attributed to the relatively high value of the malicious threshold *malValue* as well as the loyalty-based trust that leads to the decrease in the mark count. The gradual decline of the average trust value obtained by the adversaries attributes to the same phenomenon observed in the minority case. Given the high trust value bestowed upon fellow benign users, the benign users tend to discover the adversaries through their falsified recommendations, leading to a gradual tendency towards lower trust values for the adversaries.

Figure 19 shows the trust value distribution for the liberal case over all adversary populations. We observe that, in accordance with the revelations in the two sample cases above, the average trust value obtained by the benign users remains significantly high (above 0.8), even in the presence of a large number of adversaries. While their average trust value remains at complete trust when the adversaries are in the minority, it begins to drop when the adversaries go in the majority, settling between 0.8 and 0.9 after 20 rounds for 90% adversary population. Overall, we observe that the adversaries show a steady increase in the average trust value they obtained, which is however significantly higher than the average trust values obtained in the non-liberal case in Section 5.3.1.2 (c.f. Figure 13). The minor aberration in the average trust value for the case with 10% adversary population can be attributed to the low number of adversaries and their initial trust values at the beginning of the simulation. With increase in the number of adversaries in the system, we observe a more steady behavior, with a lower variance across different initial trust values.



(a) Minority Adversary Population: 20 Users, 12 Benign, 8 Adversary

(b) Majority Adversary Population: 20 Users, 6 Benign, 14 Adversary

Figure 18: Trust Value Distribution upon Collusion Attack for Liberal Trust Modification Coefficients

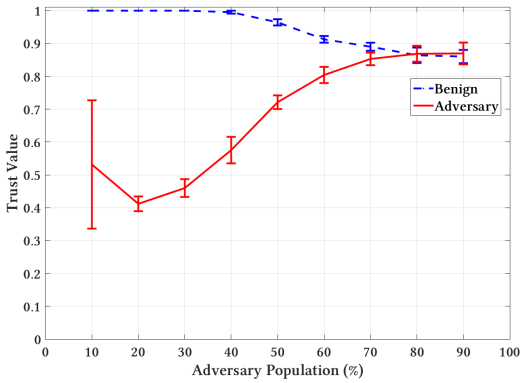
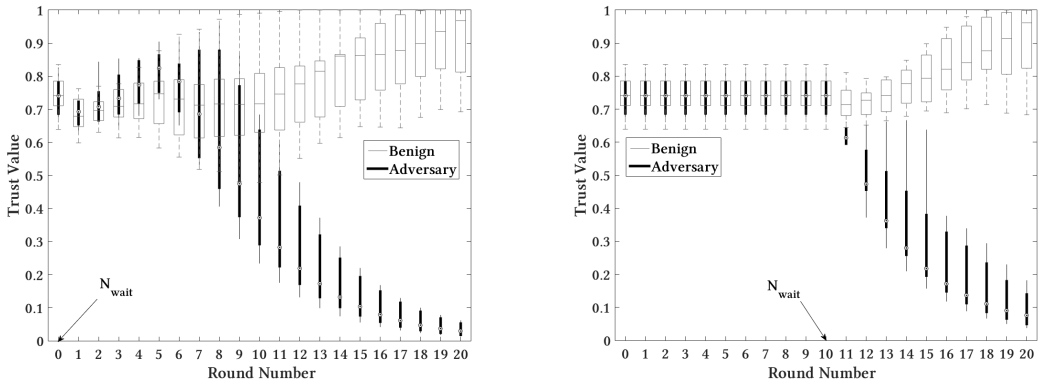


Figure 19: Trust Value Distribution upon Collusion Attack over Increasing Adversary Populations for Liberal Trust Modification Coefficients

5.3.1.6 *Influence of the Waiting Period* n_{wait}

Another parameter that plays a considerably important role in TrustCEP is the waiting period, n_{wait}, i.e., the number of rounds each user waits before applying the trust modification coefficients on the existing trust values. We analyze the influence of shorter and longer waiting periods (compared to the default setting of $n_{wait} = 4$ in the above experiments) on the trust value distribution in the system under a collusion attack. We restrict our analysis to the case with a minority adversity population of 40% to understand the implications of the waiting period. We employ the same initial trust values as in the above evaluation approaches, and analyze the average trust value obtained by the benign users and adversaries over the course of 20 rounds. For these evaluations, we considered the default values for the trust modification coefficients, as in Table 14. The results are plotted in Figure 20.

Figure 20a shows the results for the case without any waiting period, i.e., $n_{wait} = 0$. We observe that the lack of waiting period leads to an initial increase in the



(a) 20 Users, 12 Benign, 8 Adversary $n_{wait} = 0$

(b) 20 Users, 12 Benign, 8 Adversary $n_{wait} = 10$

Figure 20: Trust Value Distribution upon Collusion Attack for Different Waiting Periods

average trust value obtained by the adversaries. This is mainly caused as a result of the (positive) changes to the trust values until the malicious threshold is reached, after which the corresponding users are termed malicious and their trust values are decreased using the multiplicative decrease coefficient τ_{dec}. We observe this behavior at the end of 5 rounds, after which the trust values obtained by the adversaries declines exponentially. However, given this mild aberration in the initial few rounds, we observe that some of the benign users obtain a lower trust value for a longer period, before the users correct their initial mistake and restore trust in the benign users. This leads to an average trust value of 0.8 or higher for at least 75% of the benign users at round 18.

We present the results for the case with a larger waiting period of $n_{wait} = 10$ in Figure 20b. We observe that the average trust value obtained by the adversaries declines immediately after the waiting period elapses, considering that they shall have already exceed the malicious threshold by then. Consequently, the average trust value obtained by the benign users increases steadily after the waiting period, which attributes to the loyalty-based trust increments. The longer waiting period primarily leaves adversaries and true benign users undiscovered for a longer time interval. It is necessary to adapt the application logic in an appropriate manner to achieve a privacy-aware placement of the CEP operators.

5.3.2 *Performance Evaluation of* TrustCEP

For the performance analysis of TrustCEP, we developed a prototypical evaluation platform using user smartphones. We implemented TrustCEP in a smartphone-based distributed CEP system, which allows the users to interact directly with each other and exchange information using D2D communication. This forms the basis for the processing of operator graphs in a distributed manner. The main goal of this evaluation is to understand the feasibility and practicability of using TrustCEP in a real-life distributed CEP system, comprising user smartphones. We based our

prototypical smartphone-based evaluation platform on existing research standards towards D2D communication in the IoT, where users can collaborate with each other and process the necessary contextual information from the environment [31, 140]. This allows for reduced load on the cellular instances as well as improved latency considerations [219].

Apart from cellular communication, modern smartphones provide an array of options for D2D communication in the close neighborhood—Bluetooth[1], Wi-Fi Direct[2], NFC[3]. In our work, we choose Bluetooth, mainly due to its pervasive nature nowadays as well as its improved support for low-power and low-bandwidth data transmission (in comparison with other popular standards like Wi-Fi Direct) [74]. We designed the evaluation platform using five Google Nexus® 5 smartphones (each with Bluetooth 4.0), which are all Android-based and running an OS version of 4.3 or higher. We installed an application-based interface of TrustCEP in each of these smartphones to evaluate the performance analysis, including the battery consumption and network data usage. In order to simplify Bluetooth-based communication and the subsequent evaluation, we paired all smartphone devices beforehand.

Upon installation, each user can apply their event sensitivity levels and the required weights for the estimation of trust, based on their preferences and privacy constraints. For the initial trust vectors, we randomly generated historical interaction patterns on the two communication channels established in Section 5.2.1—calls and instant messaging—accounting for users with high and low communication intensity, as well as some without any prior interaction (corresponds to trust neutrality). In the following, we consider the smartphone devices to be an extension of the corresponding users and their privacy constraints.

5.3.2.1 *Performance Evaluation: Execution*

In the implemented smartphone application, we focus on Scenario A among the motivating scenarios described in Section 3.1, primarily due to the multitude of (possibly) sensitive events being exchanged between users. We mainly consider low-level (atomic) events generated from the microphone, GPS, and accelerometer sensors present in the smartphones. Since these atomic events are generally noise-ridden and may contain events that are not pertinent to the given scenario, we first apply a filter operator $\omega_{\phi_x}; x \in \{1, 2, 3\}$ on each of these sensors. The filtered events are then processed further using the CEP operators—aggregation, composition, derivation (ω_L, ω_M, ω_A, and ω_F)—as indicated in the corresponding operator graph, depicted in Figure 4a in Section 3.2.1. Figure 21 shows this operator graph along with the additional filter operators added to each of the producer outputs. In total, we consider 7 operators as part of the operator graph, which can be processed on the 5 devices in a collaborative manner. We modeled all events as event objects, concurring with our model for CEP introduced in Section 3.2.1. All events are appended with appropriate attributes for further processing, including a time stamp and a source node ID.

1 https://www.bluetooth.com/
2 http://www.wi-fi.org/discover-wi-fi/wi-fi-direct
3 http://nearfieldcommunication.org/

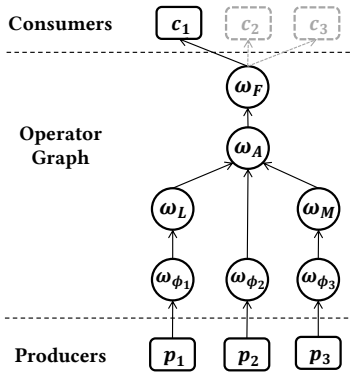


Figure 21: Illustration of the Operator Graph Used for the Performance Evaluation of TrustCEP

As part of the performance evaluation, we mainly focus on the analysis of battery consumption and network data usage during the processing of the above-mentioned operator graph. For the purpose of measuring the battery consumption, we first charge all the smartphone devices up to 100% in advance. Subsequently, we record the battery level on each smartphone device at the end of 100 execution runs of the operator graph. In order to obtain a realistic observation of the battery level change, we blocked all other communication services—especially cellular communication—and reduced the screen brightness to the minimum.

In the subsequent evaluation, we primarily focus on the analysis of battery consumption of the *leader* device, considering that it takes over the major functions of deploying an operator graph—establishment of the placement graph, placement of the operators on the respective devices, as well as trust negotiations and conflict resolution. Considering that our main motivating scenarios focus on collaboration among the devices, we compare our trust-based approach against other possible collaborative scenarios as well as scenarios without any collaboration. To this end, we consider the following use cases in our evaluation:

1. **Use Case *S*:** This case accounts for the non-collaborative scenario where all the operators are executed on a single device (of the leader) (***S*** for *single*). The leader device does not search for any other devices.

2. **Use Case *DT*:** In this case, which represents the second non-collaborative scenario, the leader device attempts to discover other devices in the environment. However, due to stringent privacy constraints, it does not trust any of them, and therefore, decides to execute the complete operator graph by itself (***DT*** for *distrust*).

3. **Use Case *CT-A*:** In this first of three collaborative use cases, we consider all devices to trust each other completely. Here, the leader device primarily takes over the processing of the atomic events (***CT-A*** for *complete trust – atomic*), while the remaining collaborating devices process the higher level events generated

by the leader device. Consequently, the leader device places the ω_{Φ_x} filter operators on itself, while delegating the remaining CEP operators in the operator graph to the other devices.

4. **Use Case *CT-H*:** This use case is similar to the previous use case **CT-A**, where all devices have complete trust towards each other. However, unlike the previous use case, the leader device primarily takes over the processing of the higher-level events by placing the corresponding operators on itself (***CT-A*** for *complete trust – higher-level operators*). The remaining devices process the atomic events by executing the filter operators.

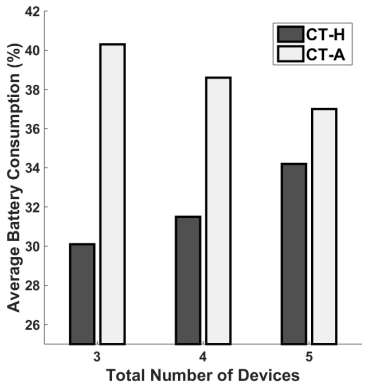
5. **Use Case *TM*:** In this use case, we employ TrustCEP for trust-based operator placement and execution (***TM*** for *trust management*). Here, the devices trust each other in accordance with their trust vectors and event sensitivity levels. They collaborate with each other based on the trust management model, by incorporating recommendation vectors and trust negotiations, to process the atomic and higher-level events generated during the execution of the operator graph. For the sake of simplicity, we do not consider the presence of any adversaries in the system.

5.3.2.2 *Performance Evaluation: Results*

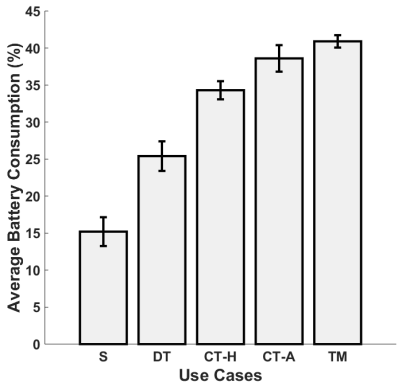
In this section, we present the results of our prototypical evaluation based on the use cases introduced above. In order to understand the implications of the various collaborative scenarios on the battery consumption and the size of network data exchanged, we vary the number of devices available and the number of operators in the operator graph for each use case.

Varying Device Availability. For analyzing the implications of D2D collaboration on battery consumption, we first executed the complete operator graph for the use cases **CT-A** and **CT-H** with 3, 4, and 5 devices in the system. To this end, we set the trust levels on all devices to the complete trust setting, $\tau = 1$. Upon observing the results of this analysis, presented in Figure 22a, we notice that the processing of atomic events is very battery-intensive compared to the that of the higher-level events. We also notice that the leader device expends more battery power in use case **CT-H** with increases in the number of collaborating devices in the system. This behavior is mainly due to the increase in the number of control messages required to place the operators on the respective devices and to maintain collaboration among them. Furthermore, we observe a drop in the battery consumption of the leader device with increases in the number of collaborating devices in use case **CT-A**. This can be attributed to the increased device availability and their adoption of the higher-level operators.

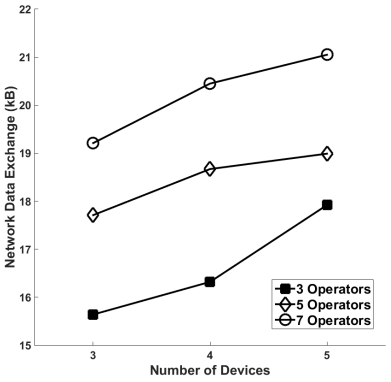
Use Cases Analysis. Subsequent to the above evaluation setup, we analyzed the effects of the 5 use cases on the battery consumption of the leader device. Unlike the previous setup, all 5 devices were considered for collaborative processing. Figure 22b shows the results obtained for each use case, where the average battery consumption of the leader device was measured. We observe that the battery consumption is least



(a) Average Battery Consumption of Leader Device over 100 Runs at Varying Device Availability



(b) Average Battery Consumption of Leader Device over 100 Runs for All Use Cases



(c) Average Bytes Exchanged per Run under Varying Number of Operators and Devices

Figure 22: Evaluation Results of the Performance Analysis of TrustCEP in Terms of Battery Consumption and Network Data Exchange

for use case **S**, where there is no communication and no collaboration with the other devices. This basically provides us with the baseline for the analysis of the other use cases. In use case **DT**, we observe that the battery consumption increases by 10.2%, mainly due to the communication costs for discovering the other Bluetooth-enabled devices in the neighborhood. Comparing the battery consumption levels in use case **TM** to that in use cases **CT-A** and **CT-H**, we observe that there is a marginal increase of 2% and 6%, respectively. Although, the operator placement can vary in each of these use cases, depending on the level of trust between devices, this marginal increase is mainly caused due to the additional recommendation messages and trust negotiation messages exchanged among the devices in use case **TM**.

Network Data Analysis. In Figure 22c, we present the results of our analysis of the number of data bytes exchanged among the devices in use case **TM** for varying numbers of devices in the environment and operators in the operator graph. For the operator graph with 5 operators, we only considered microphone and location

events, and modified operator ω_F to compose both events and derive the final context; for the operator graph with 3 operators, we only considered microphone events. We observe that increases in the number of collaborating devices and number of operators in the operator graph leads to an increase in the number of bytes exchanged. This is a logical consequence, considering that increases in devices and operators causes an increase in the number of control messages required—operator placement requests, trust recommendation messages, trust negotiations. Furthermore, the number of atomic events disseminated varies based on the different event types. Overall, the small size and low amount of data exchanged among the collaborating devices encourages the use of Bluetooth for wireless transmission, which in turn induces minimal additional battery consumption when using TrustCEP.

5.4 DISCUSSION

With the help of the above evaluations in terms of the efficacy and performance of TrustCEP, we show its applicability as a trust management model for privacy-aware placement of CEP operators in D2D environments. Its robust recommendation scheme allows for the evolution of trust by accounting for the credibility of the incoming trust recommendations. In doing so, our approach manages to prevent collusion and on-off attacks when the number of adversaries is not higher than the number of benign users in the environment. We also observed that our approach does discover adversaries, even when they are in the majority, if they do not attack all benign users.

In terms of its usage for context-aware applications, we can establish that TrustCEP introduces a very minimal additional battery consumption of less than 0.5% per operator graph execution per device, in comparison to other privacy-negligent collaborative approaches. However, we can also confirm that TrustCEP is not very suitable when the user context can be processed based on sensor data available on a single smartphone, given that the considerably lower battery consumption for single-device execution in Figure 22b. Having said that, context-aware applications in the IoT process data originating from multiple distributed information sources, which requires appropriate, privacy-aware collaboration among the devices.

In terms of the efficacy of the trust management model, the trust modification parameters, τ_{inc} and τ_{dec} must be set appropriately, so that the evolution of trust does not undergo drastic fluctuations over time. As seen in Figures 16 and 18, a wrong set of trust modification coefficients can either lead to punishing benign users or overlooking adversaries. Similarly, the waiting period n_{wait} also has a considerable influence in the functioning of the CEP system, given its influence on the time period required to detect (possible) adversaries.

Overall, in a trust-based approach for distributed CEP, it is pivotal to adapt the event sensitivity levels in accordance with the privacy constraints of the users. In doing so, only the highly trusted users receive the highly sensitive data. The trust management model may require multiple rounds to detect adversaries. This is attributed to three main factors—(i) the initial (direct) trust level of the users towards

the adversaries, (ii) the similarity of the trust recommendations to a user's own trust vector, and (iii) the attacking strategy chosen by the adversaries. It is, therefore, prudent to employ the calculated trust vectors after a set number of rounds for operator placement and execution. This also depends on the CEP application running above, given that certain events (e.g., microphone and location readings) can become more sensitive if historical information is available. One can avoid possible violations of the privacy constraints by either adapting the event sensitivity levels, accordingly, or by randomizing the event recipients in each round, until the trust levels stabilize in their value.

6 ENSURING RELIABILITY IN DYNAMIC D2D ENVIRONMENTS

In Chapter 5, we have seen how we can realize distributed CEP under privacy constraints of the users involved, by exploiting their inherent trust values towards each other. However, as discussed in Chapter 1, the user environment in the IoT is characterized by dynamic changes due to device movement, failure, or varying trust relations. Ensuring reliability (and also maintaining the established privacy constraints) in such dynamic D2D-based networks becomes increasingly difficult when dealing with resource-constrained devices, such as user smartphones (e.g., due to limited battery life or memory space).

In this chapter, we present FlexCEP, an adaptive approach to migrate CEP operators between devices by accounting for the mobility as well as the resource-constrained nature of the devices involved. FlexCEP adaptively varies the frequency of data migration as well as the amount of data to be migrated based on the execution model of CEP operators as well as the *failure probability*, $\mathcal{P}_{fail}$, of the devices involved (explained in Section 6.2). This added flexibility allows FlexCEP to achieve lower communication overhead and lower recovery times compared to traditional reliability mechanisms in CEP.

In the following sections, we first explain the main problem statement by understanding the weaknesses of the state-of-the-art approaches. We then describe our FlexCEP approach for flexible operator migration, detailing the answers to the following questions—(i) Which data must be migrated? (ii) How should these data be migrated? (iii) When should these data be migrated? Finally, we conclude the chapter by presenting and discussing some of the evaluation results of our approach.

6.1 NEED FOR A FLEXIBLE OPERATOR MIGRATION APPROACH

In CEP systems (and event processing systems in general), the flow of events dictates the inference of higher-level situational context. The loss of some events or any malfunction during the execution of an operator graph can result in falsified output events. The occurrence of false negatives—non-inference of events that actually took place—and false positives—inference of events that actually did not take place—can have severe consequences on the application at hand. For example, in Scenario B (see Section 3.1), any false negative or false positive can result in wrong speed or lane changes, effecting accidents and/or unnecessary traffic jams. Similarly, in Scenarios A and C, the non-inference of a meeting or a polluted environment can lead to embarrassing situations or health-related issues.

The main problem with distributed CEP in D2D-based networks is the possibility of network discontinuity due to node[1] movement or failure [60, 122], which in turn

1 We use the terms *node* and *device* interchangeably in the chapter.

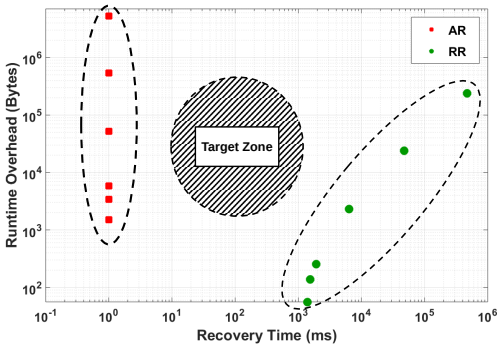


Figure 23: Comparing Approaches on Reliability in CEP

leads to the loss of events and/or falsified output events. As shown in the above scenarios, node mobility is quite prevalent in such environments. The performance of devices, like smartphones, can fluctuate due to the restricted amount of resources, such as battery life and memory space. Hence, in order to maintain reliability, it becomes vital to adapt the execution of CEP operator graphs to the dynamic changes in the environment.

Existing approaches based on active replication and roll-back recovery (see Section 2.1.2) do not address these concerns appropriately. Figure 23 shows the runtime communication overhead and recovery time values for both active replication and roll-back recovery approaches for an increasing event generation rate, R_{eg}. Recall from Section 2.1.2 that a typical active replication approach mainly accounts for horizontal dependency through redundant processing paths. In turn, it results in a low recovery time upon failure (ideally zero) but exacts increasing communication overhead and unnecessary processing resources on the backup nodes during failure-free operation, especially in relatively stationary or *quasi-stationary* scenarios. This can be seen in the figure, where the runtime overhead increases from the order of 10^3 to nearly 10^7 for $R_{eg} = 1$ event/s to 100 events/s, respectively.

On the other hand, rollback recovery approaches target vertical dependency by maintaining large buffers—sometimes in the order of GBs [189]—to allow for complete recovery. However, while they have considerably lower processing and communication overhead during runtime, they suffer from higher recovery times (and recovery communication overhead [not shown in figure]) upon failure, in the order of multiple seconds, as seen in Figure 23. Thus, such approaches effect higher number of false positives or negatives due to the possible loss of events, as we show in our evaluations in Section 6.4.

Any reliable CEP system has to account for horizontal and/or vertical dependencies between the different operators in the operator graph. Given the characteristics of D2D-based networks mentioned above, it is necessary to mitigate unnecessary processing and communication overhead, while still allowing for fast recovery mechanisms. Consequently, in our approach, we mainly target the striped region in the figure. Given the constrained view of the network, it is also important to improve

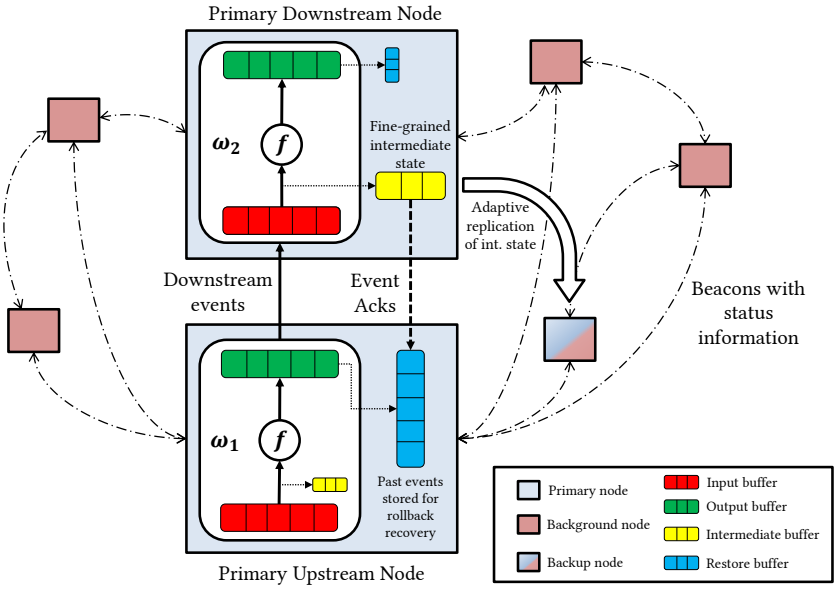


Figure 24: Overview of FlexCEP's Message Exchange and Buffer Maintenance

coordination among the devices through appropriate control messages, so that the buffer space and replicated content are updated appropriately. In turn, in our approach, we mainly address the following research questions:

RQ6.1: Which CEP operator state (i.e., the input events and the internal processing state) are required during migration to fully restore the operator graph?

RQ6.2: How can we account for both horizontal and vertical dependencies by introducing minimal overhead?

RQ6.3: How can we determine when and where to migrate operator state?

6.2 FLEXIBLE OPERATOR MIGRATION USING FLEXCEP

In this section, we describe our approach called FlexCEP for flexible operator migration[2] in distributed CEP systems. We target dynamic D2D environments with mobile and resource-constrained nodes, where reliable event processing cannot be guaranteed with just the initial placement of the operators on the available nodes. In FlexCEP, we provide a flexible and proactive approach that adheres to different network conditions, depending on the mobility of the nodes, the frequency and intensity of the generated events, as well as sudden network/node failures. We deal with horizontal and vertical dependencies by adapting our approach to the prevailing conditions, where we control the amount of operator state and the frequency of state migration among the available nodes, accordingly.

2 For the sake of convenience, we use the term *operator migration* in place of *operator state migration*, but we always imply the latter.

Figure 24 provides an overview of FlexCEP, including the messages exchanged and the buffers maintained to facilitate efficient operator migration. For a given query, we assume that the initial placement of the operator graph is performed using an appropriate placement algorithm, based on the criteria for the given application—either performance-related criteria [38, 121, 157] or based on our TrustCEP approach. We term these devices as the primary nodes on which the operator graph is executed. Each primary node executes at least one operator and sends the resulting complex events to the downstream nodes in the operator graph, until the final results reach the interested consumer node(s).

For the purpose of migration, we consider a set of background nodes that can act as backup to the primary nodes. The primary nodes keep track of each other as well as the background nodes with respect to their location, speed, movement direction, and energy resource conditions. We consider a hybrid, proactive approach, where we apply the principles of both active replication as well as rollback recovery, based on network conditions. In particular, the upstream nodes keep track of their downstream counterparts, and initiate migration based on their prediction of the location and resource availability of the corresponding devices (e.g, if they realize that they are moving away from each other). An upstream node initiates a fine-grained passive replication when it estimates the non-availability of the downstream node in the near future (e.g., an outward movement of the downstream node or depleting resource availability), by selecting one (or more) suitable backup node(s) based on the above-mentioned criteria. It also maintains a checkpointing buffer for upstream backup called the restore buffer, $B_{restore}$, in order to maintain the *relevant* events for facilitating recovery upon abrupt failures.

To this end, we answer the three research questions established in Section 6.1 with the following three contributions, respectively:

C6.1: We present a fine-grained view of the operator execution model to understand the essential components of a CEP operator and their working. In doing so, we determine the key execution stages for a given operator and ascertain the intermediate states that are relevant for replication and checkpointing (see Section 6.2.1 and Section 6.2.2).

C6.2: In order to account for the different operation states and the different dependency models, we maintain additional buffers on the nodes. In Section 6.2.3, we show how we manage the different buffers, as part of the *Buffer Management Unit* (see Section 3.3, by adapting the control and synchronization messages, appropriately.

C6.3: Finally, in Section 6.2.4, we present our predictive analysis of the available nodes using the *Failure Probability Calculator* to facilitate a proactive approach for operator migration. We show how the nodes keep track of each other by exchanging vital information about their instantaneous characteristics, in order to help detect the opportune moments for migration.

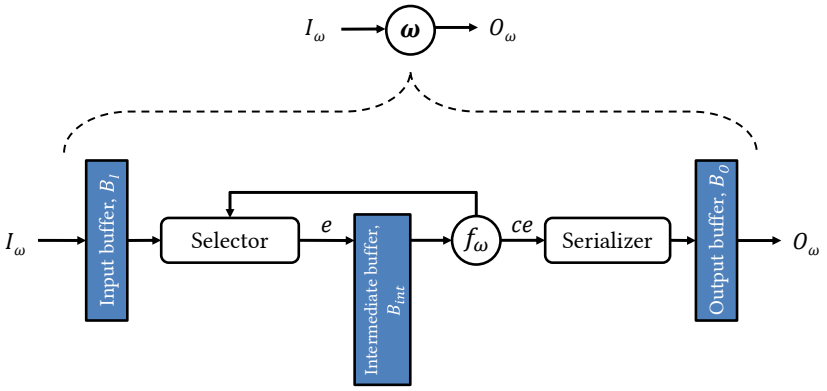


Figure 25: Illustration of a CEP Operator State Model

6.2.1 *Fine-Grained Operator Execution Model*

Our work deals with the migration of CEP operators and the involved events in order to allow for a reliable processing environment. To this end, we first understand how CEP operators handle events internally and how they produce outgoing complex events, thus allowing us to obtain a fine-grained look into the working of a CEP operator. Figure 25 depicts the internal logic of a typical operator ω used in CEP, which accepts an incoming event stream I_ω and produces an outgoing event stream O_ω.

The incoming event stream I_ω is first stored in an input buffer B_I, until the events are either selected or discarded by the internal processing logic for the given operator ω. Similarly, the complex events *ce* produced by the operator are stored in the output buffer B_O, to be delivered to the next downstream operator. The contents of the output buffer B_O are removed as and when they are acknowledged of receipt by the downstream operators.

The intermediate processing engine of an operator comprises three components—selector(s), the internal correlation function f_ω, and a serializer. The selectors are responsible for analyzing the input buffer B_I and determining the set of events e in I_ω that satisfy the criteria specified in the processing logic for the given operator ω. For example, the selector of a `WINDOW`-based operator only accepts events that occur in the specified time window for the given query. The accepted events e are said to be *consumed* by ω, so that B_I can make room for the next set of incoming events. There can be more than one selector, depending on the processing logic for a given operator. For example, a `JOIN` operator combines two (or more) incoming event streams based on a specified condition to produce a new output event stream. In this case, it contains as many selectors as the number of incoming event streams.

Alongside the input and output buffers, we consider an intermediate buffer B_{int} between the selector(s) and the internal correlation function f_ω of the operator. The intermediate buffer mainly maintains the events on which the correlation function must be applied to obtain the output complex events *ce*. The state of the selector and

the contents of the intermediate buffer depend on the processing state of the correlation function f_ω. For example, for a `WINDOW-AGGREGATE` operator, the intermediate buffer stores the events that occur in the specified time window. The corresponding correlation function f_ω applies the aggregate function (e.g., sum or average) on these events to produce the output complex events, and then updates the state of the selector accordingly, such that the contents of B_{int} shift to the next time window. We shall delve deeper into the significance of the intermediate buffer with respect to different operators in the next section (6.2.2).

Finally, the events $(ce_1, ce_2, \ldots, ce_n) \subseteq CE \in E$ produced by f_ω are appended with a sequence number and the source ID of the device by the serializer and sent to the output buffer B_O. The internal state of the serializer is basically the sequence number to be assigned to the next event produced by f_ω. In our work, we use the system timestamp (using the Unix time format in milliseconds) to set the sequence number of a given event. We assume that there exists an appropriate time synchronization algorithm [180] underlying our approach to allow for collision-free sequence numbers. Consequently, the essential components of operator state of ω at time t, $\Psi_\omega(t)$, consists of the contents of B_I, B_O, and B_{int}, as well as the processing state of f_ω (including that of the selector) and the serializer.

In general, any operator migration approach for operator recovery requires the replication of operator state between the primary and backup nodes. Different types of operators possess different levels of state. The crux of the problem lies in the amount of state information that needs to be replicated and the amount of state information that can be stored and replayed for migration. This forms the basis of our approach for adaptive operator migration, where we exploit the intermediate state of certain common CEP operators to minimize the amount of state information required for a successful migration.

6.2.2 *Intermediate State of a CEP Operator*

As indicated above, the intermediate state of a CEP operator contains information about the events that are relevant for further processing, which we store in the intermediate buffer B_{int}. To better explain the significance of the contents of this buffer, we consider the following set of common CEP operators—`FILTER`, `AGGREGATE`, `SEQUENCE`, and `JOIN`. We choose these operators to show the different possible intermediate states and their implications on the amount of data to be migrated. An exhaustive discussion of all possible CEP operators will go beyond the scope of this thesis; for a detailed list of all operators, please refer to Cugola and Margara's survey on information flow processing (IFP) [53]. The four operators are presented in Listing 1 using the Siddhi Query Language (SiddhiQL) [205].

`FILTER`. Basically, a `FILTER` operator (part of the family of single-item operators [53]) selects all the events e in the incoming event stream I_ω (stored in B_I) that satisfy a particular criterion (or many criteria), and discards the rest of the events. The output event stream O_ω then contains the selected events e. For example, one such operator can be used to accept only temperature events if the temperature value is,

e.g., greater than 30° C, as shown in Listing 1. Typically, a `FILTER` operator is used to select those events that are relevant for a more complex query. In terms of the internal functionality, the selector and serializer are effectively stateless, since the correlation function f_ω mainly sorts out the incoming event stream. Thus, the main events necessary for migration are the contents of the input and output buffers, B_I and B_O.

Listing 1: Selected CEP Operators

```
define stream temperatureStream1(roomName string, value double, time long)
define stream temperatureStream2(roomName string, value double, time long)

FILTER:
from temperatureStream1[value > 30]
select roomName, value
insert into outputStream

AGGREGATE (Average):
from temperatureStream1#window.lengthBatch(10)
select roomName, avg(value) as avgValue
insert into outputStream

SEQUENCE:
from e1 = temperatureStream1
  -> e2 = temperatureStream1[value > e1.value && time > e1.time]
  -> e3 = temperatureStream1[value > e2.value && time > e2.time]
within 10 seconds
select e1.roomName, e1.value, e2.value, e3.value, e3.time
insert all events into outputStream

JOIN:
from temperatureStream1#window.length(10) as T1
join temperatureStream2#window.length(20) as T2
on T1.value == T2.value
select T1.roomName, T2.roomName, T1.value
insert all events into outputStream
```

AGGREGATE. Often, CEP operators need to consider a certain set of events that fall in a given time period or spacial range, specifically defining the scope of their operation. An `AGGREGATE` operator selects those events in B_I that fall in the specified time window (time-based) or lie within a given number of events (length-based), and then finds the aggregate (e.g., sum, average) over those events to produce the output event stream. For one such operator, either a (fixed-size) *sliding window* is used—where the upper and lower bounds of the window advance as and when new events enter B_I—or a *batch window* is considered—where the time window always moves by the specified window size, and each event in the window is processed only once. In a time-based window, the selector compares the timestamps of each event in B_I against the specified time window and stores the selected events e in B_{int}. Following that, the correlation function f_ω calculates the average over these events and

sends the output event *ce* to B_O (through the serializer). For example, the AGGREGATE operator described in Listing 1 is used to determine the average temperature in a room, such that the temperature readings are analyzed (continuously) over a batch window of the past 10 events. Here, the current state of the content of B_{int} at a given point in time indicates the current set of events pertinent to the given operator. Consequently, this allows us to discard the irrelevant events at regular intervals, thus reducing the number of events to be transferred during migration.

SEQUENCE. A SEQUENCE operator analyzes the incoming event stream(s) within a given time window to discover inherent sequential patterns in them, where a certain set of events occur one after another in temporal order. For example, the SEQUENCE operator in Listing 1 analyzes an incoming event stream containing temperature events, and detects the sequence of three temperature readings that show an increase in the temperature value, within a time window of 10 seconds. One such burst of temperature readings can indicate that the air-conditioning system in a room is malfunctioning. Typically, it may so happen that the events arrive in a haphazard order at the operator. The correlation function f_ω adapts the functionality of the selector to the contents of the intermediate buffer B_{int}, such that the selector only permits the events e that satisfy the above sequence in the specified time window. In the above example, if an event e_1 has been detected and stored in B_{int}, the selector only permits an event e_2 or e_3 by discarding all the other events in between in the given time window. Upon detection of the required sequence (or after the time window expires), the correlation function f_ω sends the corresponding output event *ce* to the output event stream. In this case, capturing the instantaneous state of B_{int} as well as f_ω allows us to determine the relevant events for migration.

JOIN. JOIN operators allow for the merging of two (or more) incoming event streams based on certain criteria. Typically, each event in one incoming event stream I^i_ω are matched against all events in another incoming event stream I^j_ω within a given time window based on the specified condition(s) (see Listing 1). The correlation function f_ω then generates an output event for all matching event pairs. In this case, the intermediate buffer B_{int} contains the set of matched events from the event streams, that are relevant for further processing. Based on the the timestamps of the events in B_{int}, it is possible to discard the irrelevant events in all incoming event streams.

Apart from the single-item FILTER operator, the remaining operators possess significant intermediate state, which becomes increasingly important for operator migration in resource-constrained and dynamic environments. Instead of replicating/logging entire sets of events to maintain reliable execution of CEP operator graphs, the extraction of the intermediate state—especially the intermediate buffer B_{int}—cuts down the transfer/storage of irrelevant events and thus, reduces computation intensity on the primary as well as the backup nodes, considerably. In turn, the backup nodes may resume processing where the primary nodes left off.

6.2.3 *Adaptive Buffer Synchronization*

Having analyzed the significance of the intermediate state for different CEP operators, we now present our approach for maintaining the internal buffers involved in a FlexCEP node, in order to facilitate efficient operator migration. Our approach entails an adaptive content synchronization mechanism among the different internal buffers—the input buffer B_I, the output buffer B_O, the intermediate buffer B_{int}, as well as the restore buffer $B_{restore}$ (introduced above)—on the primary and backup nodes, depending on the prevailing network conditions. The main steps of this approach are detailed in Algorithm 3.

Algorithm 3 : Adaptive Buffer Synchronization

Description : Algorithm for the management of the intermediate buffer B_{int} on the primary downstream node and the restore buffer $B_{restore}$ on the primary upstream node

Variables : $R_{int} \leftarrow$ rate of incoming event stream
$\phi_{restore} \leftarrow$ restore buffer synchronization factor
$R_{min} \leftarrow$ minimum rate of acknowledgment

```
1  function receiveEvents(⟨InputEventStream⟩)
2      for each e ∈ ⟨InputEventStream⟩ do
3          B_I.add(e);
4      t_1 ← currentTime();
5      for each e ∈ consumedBySelector() do
6          B_I.remove(e);
7          if e.toBeProcessed() then
8              B_int.add(e);
9      t_2 ← currentTime();
10     R_int ← B_int.size() / (t_2 − t_1);
11     sendAckToUpstreamNode(B_int, φ_restore, R_int, 0);
12     saveStateSnapshot();
13 function sendAckToUpstreamNode(⟨InBuffer⟩, φ, R, R_min)
14     for i = 1 to InBuffer.size() do
15         ackWait ← 0;
16         if i ⩾ (max(φ · R, R_min) + ackWait) then
17             sendAck(InBuffer.getSeqNo(i));
18             ackWait ← i;
19 function receiveAck(seqNo)
20     for each e ∈ B_restore do
21         if seqNo ⩾ e.getSeqNo() then
22             B_restore.remove(e);
```

As mentioned in Section 6.2.1, a typical CEP operator accepts incoming event streams in the input buffer B_I and inserts the produced complex events in the output buffer B_O. Once the events are *consumed* by the selector—either selected for further processing or discarded upon not meeting the specified criteria—they are removed from the input buffer to make room for the next set of events in the incoming event stream(s) (Lines 5–6). The events selected for further processing are inserted into the intermediate buffer B_{int} (Lines 7–8). The contents of the output buffer are transferred to the input buffer of the subsequent downstream operator(s)[3], and are evicted from the output buffer upon receipt of the corresponding acknowledgments from the successors. This functionality of the input and output buffers is well established in current CEP systems to ensure proper transfer of events through the operator graph [116].

Typically, in existing literature on active replication and rollback recovery [22, 65, 86, 116, 217] in order to minimize the control communication overhead, the rate of acknowledgments R_{ack} is adapted to the rate of event reception R_{er}, i.e., the rate at which the events enter B_I in the downstream node. In doing so, $R_{ack} = max(\phi_{ack} R_{er}; R_{min})$, where the constant $\phi_{ack} \in (0, 1]$ is the *cumulative acknowledgment factor*, and R_{min} is the minimum rate of acknowledgment that has to be maintained for reliable communication. Consequently, the receipt of every $\lfloor \frac{1}{\phi_{ack}} \rfloor^{th}$ event is acknowledged by the downstream node as a cumulative acknowledgment for the past $\lfloor \frac{1}{\phi_{ack}} \rfloor$ events (of course, subject to R_{min}).

However, considering that our approach deals with a resource-constrained and mobile environment, we refrain from using acknowledgments for event reception. In turn, as soon as an event is sent out of the output buffer B_O of an upstream node to the input buffer B_I of the downstream node, the corresponding event is removed from the output buffer to make space for the next set of events. In order to allow for efficient recovery purposes, we instead introduce an additional buffer on each node, called the restore buffer $B_{restore}$.

The restore buffer stores all the events that enter the output buffer B_O but are not yet consumed by the internal processing logic of the downstream operator. Based on the events that are consumed for further processing by the downstream operator, i.e., based on the instantaneous contents of the corresponding B_{int}, the contents of $B_{restore}$ on the upstream operator are updated. Data synchronization acknowledgments, much akin to event acknowledgments described above, are sent from the downstream node to the upstream node, so that the irrelevant events as well as the events in B_{int} on the downstream operator can be evicted from the corresponding $B_{restore}$ (Lines 20–22). If we define the rate at which events enter B_{int} as R_{int}, then the rate of restore buffer synchronization between the downstream and upstream nodes, $R_{restore}$, can be defined as $R_{restore} = \phi_{restore} R_{int}$, where $\phi_{restore} \in (0, 1]$ is the *restore buffer synchronization factor* (Line 11). The restore buffer acts as each operator's

3 For the sake of simplicity, in the ensuing discussion, we restrict the number of primary upstream and downstream nodes (and therefore, the number of operators that run on each node) to one each. This also applies for backup and background nodes. However, the concepts developed can be applied to multiple-node scenarios.

log for operation checkpoints, thus allowing for rollback recovery in case of abrupt node/network failures. The main premise behind this design decision is the maintenance of only those events that are relevant for the execution of the prevailing context query, and reducing the amount of data to be stored on one device for the purpose of migration.

With the requisite input events available in $B_{restore}$ on the upstream node, the main contents of Ψ_ω on the downstream node at a given point in time t include the contents of B_{int} as well as the processing state of f_ω and the serializer. Additionally, we also include the contents of $B_{restore}$ of the downstream node itself in Ψ_ω to account for the outgoing event stream. We take a *snapshot* of Ψ_ω at regular intervals, such that this snapshot can be used to restore the operator ω upon migration by instantiating ω with the exact state on the backup node. This allows us to resume operation on the backup node as it were on the primary node, by using the contents of $B_{restore}$ on the upstream node as the initial input event stream.

The use of the above buffers allows us to store and transfer the required amount of data (events and operator state) to the backup nodes to complete operator migration successfully. However, the question of when and where to transfer the data is an important one. Updating the state of an operator Ψ_ω on the backup nodes can effect a large communication overhead. It is essential to determine the moments in time when one such migration is necessary and also, to determine the right candidates that can act as backup nodes, depending on the device and network conditions.

6.2.4 *Mobility-Aware Operator Migration*

As part of FLEXCEP, we devise a proactive approach for operator migration that accounts for the dynamic changes in D2D-based networks mentioned in Section 6.1, such that the communication and processing overhead is kept to the required minimum, while still facilitating reliable processing. Using estimates of the movement and resource availability of the available nodes, we establish how backup nodes are chosen, when they are activated, and how operator state is replicated/synchronized between the primary and backup nodes. Consequently, we present an approach that allows for flexible execution of an operator graph, depending on the prevailing network conditions (see Algorithm 4).

Each node captures and stores status information about its location (e.g., GPS coordinates) as well as its resource characteristics, i.e., battery level and memory space, at regular intervals of time. This set of information is exchanged among the nodes using the *beaconing* component (see Section 3.3), by sending the information as beacons at regular intervals of time. The rate of beaconing, R_{beacon}, can be varied to handle different mobility patterns. This helps the primary nodes keep track of the other nodes in the processing chain, thus allowing for proactive measures in case of any aberrations. The status information of the other nodes is used to predict their failure probability in the near future. We define the failure probability of node i for the time period $(0, t)$ as $\mathcal{P}^i_{fail}(t)$, which chiefly depends on three factors—the probability of moving out of range, $\mathcal{P}^i_{OOR}(t)$, based on the relative velocity of the

Algorithm 4 : Mobility-Aware Operator Migration w.r.t. node j

Description : Algorithm for processing incoming beacons and calculating the failure probability $\mathcal{P}_{fail}$ of the corresponding node. If $\mathcal{P}_{fail} \geqslant failValue$, the node initiates migration to closest viable backup node.

Variables : $nodeID \leftarrow$ unique ID of a node
$t_{capture} \leftarrow$ time instant at which the beacon was captured
$\langle loc \rangle \leftarrow$ location data in X and Y coordinates
$bat \leftarrow$ current energy value as percentage
$mem \leftarrow$ current memory capacity as percentage
$\mathcal{P}_{OOR} \leftarrow$ probability of moving out of range
$\mathcal{P}_{res} \leftarrow$ probability of resource depletion
$\mathcal{P}_{crash} \leftarrow$ probability of abrupt crash
$w_{OOR}, w_{res}, w_{crash} \leftarrow$ weighting factors
$backupLimit \leftarrow$ degree of horizontal dependency

```
1  function receiveBeacon(nodeID=i, t_capture=t_c, ⟨loc^i⟩, bat^i, mem^i)
2      locLog⟨i, t_c⟩ ← ⟨loc^i⟩;
3      P^i_OOR(t_c) ← calcProbOOR(locLog⟨i, t_c⟩, locLog⟨j, t_c⟩);
4      P^i_res(t_c) ← calcProbResDepletion(bat^i, mem^i);
5      P^i_crash(t_c) ← calcProbCrash(n^i_crash(t_c));
6      P^i_fail(t_c) ← calcProbFail(P^i_OOR(t_c), P^i_res(t_c), P^i_crash(t_c), w_OOR, w_res, w_crash);
7      for each node ∈ ⟨backgroundNodes⟩ do
8          if node.getProbFail() < passValue then
9              backupCandidates.add(node);
10         else
11             backupCandidates.remove(node);
12     backupCandidates.sortByLocation(locLog⟨j, t_c⟩);
13     if downstreamNode.getProbFail() ⩾ failValue then
14         for k = 1 to backupCandidates.size() do
15             if k ⩽ backupLimit then
16                 backup.add(backupCandidate(k));
17         initiateMigration(⟨backup⟩, downstreamNode, backupLimit);
```

node with respect to the others; the probability of resource depletion on the node, $\mathcal{P}^i_{res}(t)$, based on the instantaneous status of node energy and memory capacity; as well as the crash probability, $\mathcal{P}^i_{crash}(t)$, which depends on the number of past abrupt crashes. Each node estimates the failure probability of the other nodes based on the status information obtained via beaconing as well as past experiences with respect to abrupt crashes. The failure probability estimated by node j with respect to node i for the time period $(0, t)$ (i.e., the previous estimate was made at $t = 0$) is given by,

$$\mathcal{P}^i_{fail}(t) = min\left(w_{OOR} \cdot \mathcal{P}^i_{OOR}(t)|_j + w_{res} \cdot \mathcal{P}^i_{res}(t) + w_{crash} \cdot \mathcal{P}^i_{crash}(t);\ 1\right) \tag{7}$$

where the parameters w_{OOR}, w_{res}, and w_{crash} are weighting constants. such that $w_{OOR} + w_{res} + w_{crash} = 1$ (Line 6). Here, $\mathcal{P}_{OOR}$ is subject to the relative distance and velocity between the nodes; each node can therefore have a different failure probability with respect to the other nodes.

The probability of moving out of range $\mathcal{P}_{OOR}$ depends on the (relative) speed of the node and (relative) angle of movement, as calculated by other nodes based on the location information exchanged. If $(loc^i_x, loc^i_y)|_{(t=t_1)}$ and $(loc^j_x, loc^j_y)|_{(t=t_1)}$ are the location coordinates of nodes i and j, respectively, at $t = t_1$, then the slope of the imaginary line connecting them is given by $m_{i,j}(t_1) = \frac{loc^j_y - loc^i_y}{loc^j_x - loc^i_x}$. By comparing the slopes of the connecting line at a different instant of time $t = t_2$, we can obtain the relative angle of movement θ of node i with respect to node j, as in Equation (8). The instantaneous speed v of node i is also calculated based on the location coordinates at the two time instants.

$$\begin{aligned} \theta(t_2) &= tan^{-1}\left(\frac{m_{i,j}(t_2) - m_{i,j}(t_1)}{1 + m_{i,j}(t_1)m_{i,j}(t_2)}\right) \\ v(t_2) &= \frac{\sqrt{\left(loc^i_x(t_2) - loc^i_x(t_1)\right)^2 + \left(loc^i_y(t_2) - loc^i_y(t_1)\right)^2}}{t_2 - t_1} \end{aligned} \tag{8}$$

We consider a node to be have a higher probability of moving out of range if the (instantaneous) angle $\theta \geqslant \frac{\pi}{2}$ (i.e., 90°) or if it fluctuates over time [150]. We also ascertain that the node's (relative) speed is directly proportional to its probability of moving out of range. Basically, each measuring node observes the instantaneous difference in the speed of the other nodes with respect to its own. Considering that the probability of a node moving out of range increases with increases in the relative difference in speeds—with the upper limit at 1—we employ an exponential model to depict the probability of a node moving out of range.

Consequently, since changes in the angle of movement as well as the speed are independent of each other (especially on highways with multiple lanes), we define the probability of node i moving out of range of node j in time period (t_1, t_2), $\mathcal{P}^i_{OOR}(t_2)|_j$ as shown in Equation (9) (Line 3).

$$\mathcal{P}^i_{OOR}(t_2) = min\left(1 - \frac{1}{\alpha^{v_i(t_2) - v_j(t_2)}} + min\left(\frac{\theta^i(t_2)}{\frac{\pi}{2}};\ 1\right) \cdot \left(1 - \frac{1}{\alpha^{\frac{d\theta^i(t_2)}{dt}}}\right);\quad 1\right) \tag{9}$$

The term α represents the type of exponential curve applicable for the calculation of the out-of-range probability. This depends on the real-world communication range values for the application case at hand. For example, in Scenario B in Section 3.1, the point at which a vehicle exits the range of the other vehicles can be used to determine the type of curve that applies for the given scenario using logistic regression techniques[4].

Similarly, if we define $bat^i(t_1)$ and $mem^i(t_1)$ as the percentage of battery power and the memory space available on node i at time $t = t_1$, respectively, we can define the probability of resource depletion for the period (t_1, t_2) as in Equation (10) (Line 4).

$$\mathcal{P}^i_{res}(t_2) = \left(\frac{1}{bat^i(t_1) + \epsilon} + \frac{1}{mem^i(t_1) + \epsilon} \right) \tag{10}$$

Note that ϵ accounts for a negligibly small positive value to avoid possible division-by-zero errors. If this applies, the resulting value of probability becomes very high (tending to infinity), which we then substitute with 0 for further calculations.

And finally, the crash probability $\mathcal{P}^i_{crash}(t_2)$ of node i for the above period is given by $\mathcal{P}^i_{crash}(t_2) = \left(1 - \alpha^{-n^i_{crash}(t_1)+\epsilon}\right)$ (Line 5). Here, $n^i_{crash}(t_1)$ stands for the number of abrupt crashes node i has had until time instant $t = t_1$. We define abrupt crashes as occasions where a node is assumed to be no longer reachable, either due to lack of acknowledgments or due to lack of status information beacons. Again, the term α depends on real-world scenarios which allow for the estimation of the logistic regression curve between the probability values and the number of crashes.

An upstream node in the operator graph estimates the failure probability of the corresponding downstream node using Equation (7) to determine when the migration approach should be initiated. The upstream node also selects a background node that can act as backup if the primary downstream node fails. Based on the status information beacons from the background nodes, each primary node first selects the background nodes that have a failure probability lower than a threshold called *passValue* (Lines 7–11). If a background node's failure probability goes beyond the pass threshold, they are removed from the list. Among the shortlisted background nodes, the primary upstream node selects the closest node that has the least failure probability and notifies it about the corresponding primary downstream node (Lines 12–16). The number of backup nodes selected can also be controlled based on the degree of horizontal dependency required for the application at hand, defined by the *backup limit* (Line 15).

When an upstream node detects that the failure probability of the corresponding downstream node is estimated to be above a threshold called *failValue*, it informs the backup node to contact the downstream node and initiate the passive replication phase (Line 17). This mainly entails the replication and synchronization of the intermediate state of the downstream node on the backup node, so that the backup node can take over operations immediately. The *backup synchronization rate*, R_{backup}, depends on the size of the intermediate buffer of the corresponding operator run-

4 https://en.wikipedia.org/wiki/Logistic_regression

ning on the downstream node. The downstream node synchronizes its state with the chosen backup node at regular intervals, to account for any abrupt crashes on its part. In this thesis, we set $R_{backup} = R_{restore}$, so as to avoid any possible inconsistencies in operator state migration. Finally, when switching to the backup node for subsequent processing, the contents of $B_{restore}$ on the primary upstream node are transferred to B_I on the backup node. This two-step process—(i) entailing passive replication of intermediate state from the downstream node and rollback recovery of the relevant events on the upstream node, and (ii) the failure probability measure—allow us to control the working of the backup node according to the application's needs.

6.3 EVALUATION SETUP

Before we proceed to the evaluation of our approach, we need to define the evaluation criteria based on which we analyze our approach. In the following, we describe our evaluation setup and then explain the procedure followed to simulate the evaluation environment and to obtain the results as per the established criteria.

6.3.1 *Evaluation Criteria*

In our evaluations, we compare our FLEXCEP approach against the two main categories of reliability approaches in related literature—*active replication* and *rollback recovery / upstream backup*. In addition to these two approaches, we also consider an *improved* active replication approach that combines the properties of rollback recovery and active replication. For our evaluations, we implemented the above approaches as below:

Active Replication (AR). Active replication entails the replication of events on one (or more) redundant paths in order to support reliable processing in case one of the paths fails due to node failure, as shown in Figure 3a in Section 2.1.2. For the sake of comparison, we only consider one redundant path (i.e., with one backup node) alongside the primary processing path in our evaluations. The primary upstream node transmits all output events to both the primary downstream node as well as the backup node. In return, the primary downstream node as well as the backup node acknowledge reception of the events by sending back the sequence number of the received events.

Rollback Recovery (RR). For rollback recovery, we consider an additional buffer on the primary upstream node that holds all output events that have been sent to the primary downstream node. We consider a traditional rollback recovery approach without any checkpointing, so as to understand the consequences of allowing for fail-safe reliability. The downstream node sends acknowledgments to the upstream node at regular intervals based on the *acknowledgment threshold* (as described in Section 6.2.3). If the upstream node does not receive acknowledgments for a certain period (based on the established acknowledgment rate), it assumes the downstream node to have failed, and therefore initiates rollback recovery by transferring all events

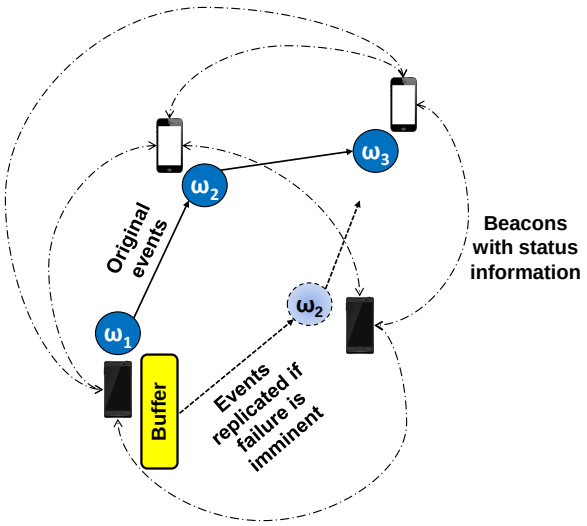


Figure 26: Illustration of the *Improved* Active Replication Approach (iAR Used in the Evaluations)

in the buffer to the backup node (see Figure 3b in Section 2.1.2). Note that the CEP operator on the upstream node is blocked for the duration of the restore process.

Improved Active Replication (iAR). Figure 26 illustrates the *improved* active replication approach, where we combine the benefits of active replication and rollback recovery based on the beacons exchanged between the nodes (as also seen in FlexCEP). Unlike the typical active replication case, only the primary processing path is used at the beginning, leaving the backup node in standby mode (passive) during failure-free runtime operation. If the primary upstream node observes a failure probability above *failValue* with respect to the primary downstream node, it initiates rollback recovery by replaying all its current output events to the backup node, and continues to use both processing paths until the primary downstream node disappears/fails. In this way, the backup node is only engaged when the primary upstream node suspects an imminent failure in the primary processing path. However, unlike our approach, this approach does not consider the processing state of the operators. We include this approach in our evaluations to evaluate the consequences of a simple combination of the two traditional approaches, without accounting for operator state information.

The main goal of our evaluation is to investigate the performance of the above approaches against our approach with respect to the following criteria:

Communication Overhead. We define the communication overhead as the number of messages exchanged between the nodes, excluding the events transferred from one node to another for processing the given query. In our evaluations, we differentiate between runtime communication overhead and recovery overhead to understand the effects of failure-free and failure-ridden operations. We define runtime overhead

as the control messages exchanged between the nodes during normal failure-free operation, which include—(i) messages exchanged to keep track of other nodes (e.g., beacons, acknowledgments), and (ii) messages sent to the potential backup nodes for reliability reasons (e.g., snapshots, event replication). Recovery overhead, on the other hand, includes all messages exchanged between the nodes to recover after a failure (or a possibly impending failure) has been recognized, typically sent by the primary upstream node to the backup node to restore the processing state.

Recovery Time. The recovery time is the amount of time required by the CEP system to restore the normal processing state after a failure occurs. Basically, the recovery time of any reliability approach has three main factors—(i) the amount of time required for the upstream node to detect failure of the downstream node; (ii) the amount of time required to start and set up the required operator on the backup node; and finally, (iii) the amount of time required to restore the state of the backup node operator to that on the failed node. As mentioned above, the primary upstream node gets blocked during recovery until the required set of events (in the corresponding buffer) are sent to the backup node. We calculate the recovery time based on the time taken for the backup node to start disseminating non-duplicate events, such that the consumer node (or the next downstream node) can continue processing the generated events as they arrive.

Duplicate Count. For each of the above approaches, we also count the number of duplicates arriving at the consumer node, based on the events sent by the primary downstream node and those sent by the backup node. We do so by comparing the content as well as the sequence numbers of the events received at the consumer node.

Event Loss. In order to verify the correctness of the events received at the consumer node, we compare them against the expected set of events in terms of their expected timestamp (in case of normal operation) and their event value. Any aberrations in the set of output events with respect to the expected set are considered as event losses, since they normally result from the loss of certain intermediate events. Furthermore, any events that are not processed by the primary upstream node due to the blockage during event restoration are also considered as lost events.

6.3.2 *Evaluation Environment*

Our CEP processing platform is based on the open-source CEP engine called Siddhi [205]. Siddhi allows for a fast and multi-threaded runtime environment to execute CEP operators by analyzing event streams in a continuous manner. Each node in our distributed CEP environment executes a separate Siddhi instance to execute the operator placed on it. Given the open-source nature of the Siddhi CEP engine, we can access the intermediate events inside the operators and extract the necessary information to facilitate adaptive buffer management in our approach.

We developed an experimental CEP system based on the Java programming language, and used Siddhi as the underlying CEP library. Each of the nodes in the distributed CEP system were implemented as Java applications, such that they each ran on a separate *Java Virtual Machine* (JVM). We deployed the nodes on a high-end server,

such that each node obtained a RAM space of 2 GB and a compute unit equivalent to a 2.1 GHz processor, which corresponds to the (minimum) specifications available on most modern smartphones on the market nowadays. Each node communicates with the other nodes using sockets. To keep the amount of information exchanged between nodes to a minimum, we employed *User Datagram Protocol* (UDP) as the underlying transport protocol. By restricting the deployment of the nodes to a single server, we mainly focus on evaluating the cost of migration in terms of communication overhead and recovery time, instead of dealing with environmental delay and loss factors. An on-field evaluation of our approach is the subject of future work in this field.

In order to simulate a mobile environment, each node obtains a set of mobility traces, which include the location coordinates and the corresponding timestamp for the given node. This information is also used to send out beacons at regular intervals of time, so as to inform the other nodes about one's status. For the mobility traces, we considered the open-source trace set provided by Gramaglia et al. [80], based on vehicular data on a set of highways around the city of Madrid in Spain. These traces include the location information of the vehicles—captured at a granularity of 500 ms—which enter and leave a fixed road segment of around 10 km. Using these traces, we can simulate Scenario B in Section 3.1 as part of the mobile evaluation scenario.

6.3.3 *Evaluation Procedure*

Our approach FlexCEP supports reliable CEP in a dynamic D2D environment, where the devices involved can have different degrees of mobility, from stationary to highly mobile. In our evaluations, we consider two different environments to account for these extremities—(i) we consider a quasi-stationary environment where the devices do not move over the course of the execution of an operator graph, but may incur abrupt crashes given their resource-constrained nature, and (ii) we consider a highly mobile automotive environment, where the execution of an operator graph can get disrupted in case one of the processing devices leaves the processing environment. In both the cases, using the FlexCEP approach, devices keep track of each other using the beaconing service.

For the purpose of our evaluations, we consider a simplified five-node D2D network as shown in Figure 27, consisting of the following nodes—(i) a producer node that acts as the event source, (ii) a consumer node that receives the processed events at the end of the operator graph, (iii) a primary upstream node and a primary downstream node, where the initial placement of the operators in the operator graph has been made, and (iv) a backup node that is in place for recovery purposes.

Evaluation Query and Operator Graph

To demonstrate the influence of intermediate operator state, we consider a query involving a `WINDOW-AVERAGE` operator as part of our evaluations (see Code 2). As part of the corresponding operator graph, the primary upstream node executes a

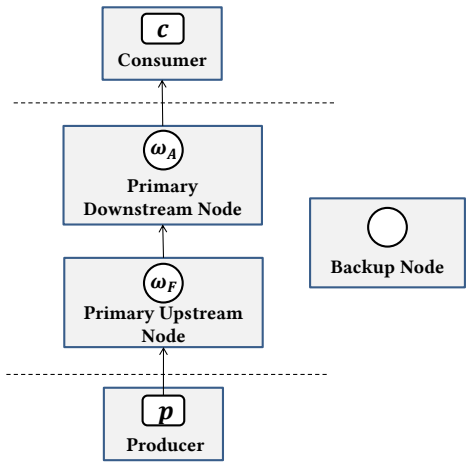


Figure 27: Evaluation Setup and Corresponding Operator Graph

`FILTER` operator (`Query1`, ω_F), in order to filter out unnecessary events from further processing based on a certain `THRESHOLD` (e.g., to filter out bad sensor output or outliers). The primary downstream node then executes a `FILTER`-based `WINDOW-AVERAGE` operator (`Query2`, ω_A) to output the average value of the relevant incoming events (`SENSOR_TYPE`) at regular intervals (based on `WINDOW_SIZE`) to produce the output events that the consumer node in the given scenario is interested in.

Listing 2: Evaluation Query

```
define stream inputStream(sensor string, reading double)

Query1 (ωF):
from inputStream[reading > THRESHOLD]
select sensor, reading
insert into outputStream

Query2 (ωA):
from outputStream[sensor == 'SENSOR_TYPE']#window.lengthBatch('WINDOW_SIZE')
select sensor, avg(reading) as newReading
insert all events into consumerStream
```

The given query can be used in the motivating scenarios described in Section 3.1 as follows: In Scenario A/C, Walt is interested in knowing the average sound levels in the concerned environment, by aggregating across a set of microphone readings at regular intervals. The primary upstream node receives events from different sensors and filters out the unwanted events. The primary downstream node then *selects* the pertinent events and finds the average over them. Similarly, in Scenario B, Walt is interested in obtaining the average speed information of a road section ahead of him, but only in the lane he is currently driving on. In this case, after the primary

Table 15: Parameters Used for the Performance Evaluation of FLEXCEP

Parameter	Value
Number of input events	20, 50, 100, 1000, 10000, 100000
Event generation rate, R_{eg} (events/s)	1, 2, 10, 100, 1000
Window size	2, 4, 8, 16, 32, 64
Cumulative acknowledgment factor, ϕ_{ack}	1
Restore buffer synchronization factor, $\phi_{restore}$	2/WINDOW_SIZE
Failure threshold, *failValue*	0.7
Beacon sending rate, R_{beacon} (beacons/s)	1, (2)

upstream node filters out the bad sensor data, the primary downstream node *selects* the events pertinent to Walt's driving lane and provides him with the average speed information at regular intervals.

Before the start of our evaluations, we assume that the operators, corresponding to the queries `Query1` and `Query2`, have been placed on the primary upstream and downstream node (based on an operator placement algorithm). For the purpose of evaluating the performance of our approach against that of the existing state of the art approaches, we only consider cases where the primary downstream node fails at an unknown instant of time (either abruptly or by moving out of range). We then evaluate the costs of supporting reliability for FLEXCEP as well as each of the above-mentioned approaches in terms of the established evaluation criteria.

Independent Evaluation Variables

Table 15 shows the set of independent variables pertinent to the evaluation of our approach. We vary the number of events to be processed as well as their rate of generation to evaluate the performance of the approaches for varying degrees of event load. We created a set of synthetic event traces to simulate different event loads, by varying the number of events generated from 20 to 100000. Each event contains a data field, comprising a string for the event name (e.g., sensor type) and the corresponding data value (e.g., sensor reading). Further, each event also has a timestamp value, which is indicative of the time it was created as well as its sequence number. Finally, additional boolean fields are included to indicate whether the event has been acknowledged or consumed. In total, each event varies between 200 and 208 bytes, depending on the string length of the event name.

For the quasi-stationary scenario, each of these traces are used as input with different values of the event generation rate, R_{eg}, from 1 event/s to 100 events/s. Furthermore, to reduce the evaluation design space, we also set the length of the operator window in accordance with the number of events per run, varying from 2 to 64. The dependencies between the number of events, R_{eg}, and the window size for the quasi-stationary scenario are shown in Table 16.

Table 16: Dependencies Between the Evaluation Variables of FLEXCEP in the Quasi-Stationary Evaluation Scenario

Number of Events	R_{eg} **(events/s)**	**Window Size**
20	1	2
50	2	4
100	10	8
1000	10	16
10000	100	32
100000	100	64

Table 17: Dependencies Between the Evaluation Variables of FLEXCEP in the Mobile Evaluation Scenario

Number of Events	R_{eg} **(events/s)**	**Window Size**
100	1	4
1000	10	8
10000	100	16
100000	1000	32

In the traditional approaches of AR and RR, the rate of acknowledgments corresponds directly to the rate of incoming events. For the purpose of facilitating fail-safe event processing, we consider both these approaches with a cumulative acknowledgment factor $\phi_{ack} = 1$, such that every incoming event is acknowledged immediately. For the FLEXCEP approach, we considered the restore buffer synchronization factor $\phi_{restore} = 2/\texttt{WINDOW\_SIZE}$, such that the primary downstream node sends an acknowledgment to the primary upstream node whenever the number of *consumed* events equals half or full of the window size. In turn, the primary upstream node receives two acknowledgments for each window-length of consumed events. This allows us to analyze the importance of the intermediate buffer in our approach. All acknowledgments have a fixed size of 8 bytes, corresponding to the long Unix time format in milliseconds.

In our approach and in IAR, each node sends beacons with status information to the other nodes at regular rates of $R_{beacon} = 1$ *beacon/s* (unless stated otherwise). Each beacon contains the following fields: `nodeID`, `timestamp`, `locationX`, `locationY`. To reduce the complexity of the evaluations, we did not consider energy and memory readings as part of the beacons. Also, considering that each event trace set is executed independently, we do not consider the abrupt crash factor n_{crash} in our evaluations. Consequently, each beacon has a total size of 56 bytes. Upon receiving a beacon, each node calculates the failure probability of the corresponding sending node with

respect to its status information, as per the algorithm described in Algorithm 4. A lack of 2 beacons in succession is considered as a sign of failure.

In the quasi-stationary evaluation scenario, we consider different instants of time to simulate an abrupt failure of the primary downstream node. We consider two time instants for node failure: an early failure which occurs after 30% of the input events has arrived, and a late failure which occurs after 80% of events has arrived. This allows us to analyze the effects of abrupt node failure on the recovery overhead and recovery time for each approach, and compare the runtime overhead in each case to the case with failure-free operation.

For the mobile evaluation scenario, we allowed the primary downstream node to fail once its measure of the failure probability of the primary upstream node goes beyond 0.99, considering that this indicates an out-of-range condition on its part. We chose the vehicular traces for the three CEP nodes in such a way that the primary downstream node moves away after a certain time, while the primary upstream node and the backup node remain in range for the duration of the execution of the given operator graph. Consequently, in Equation 9, we choose $\alpha = 2$, such that the nodes are assumed to be moving out of range when their relative speed increases beyond 10 km/h. In order to measure the effects of different event loads on the CEP system before the primary downstream node fails (i.e., moves out of range), we changed R_{eg} and window size for the different event trace sets, as seen in Table 17.

6.4 EVALUATION RESULTS

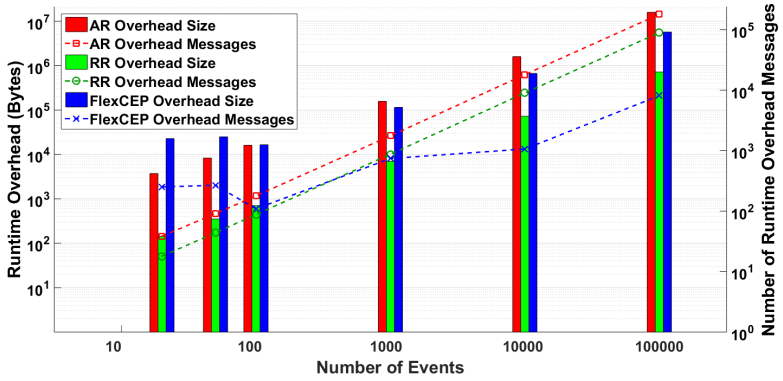
We now present the results of our evaluation based on the evaluation setup described above. We divide our evaluation into two scenarios: one in a quasi-stationary environment and another in a mobile environment. For each of these evaluations, we ran the operator graph for three different event types and present the results as the average over the individual runs.

6.4.1 *Performance Analysis in Quasi-Stationary Environments*

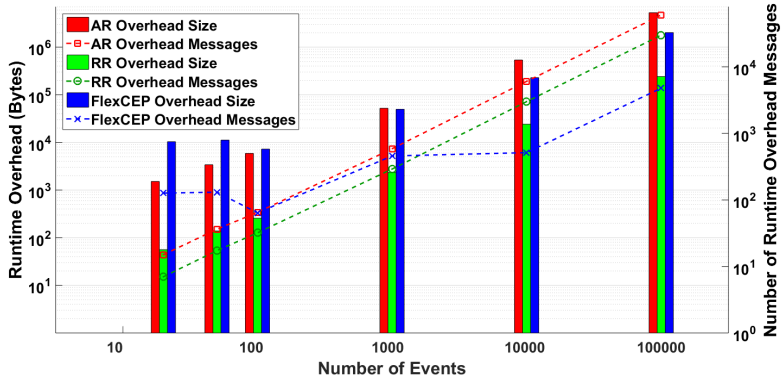
In this section, we focus on the comparison among the approaches AR, RR, and our approach FLEXCEP with respect to their performance in case of abrupt failure on the part of the primary downstream node. As part of FLEXCEP, given the quasi-stationary nature of the environment, we used a set of constant location traces as the status information for beaconing. In the following, we shall present our results with respect to the evaluation criteria described in Section 6.3.1.

Communication Overhead: Runtime

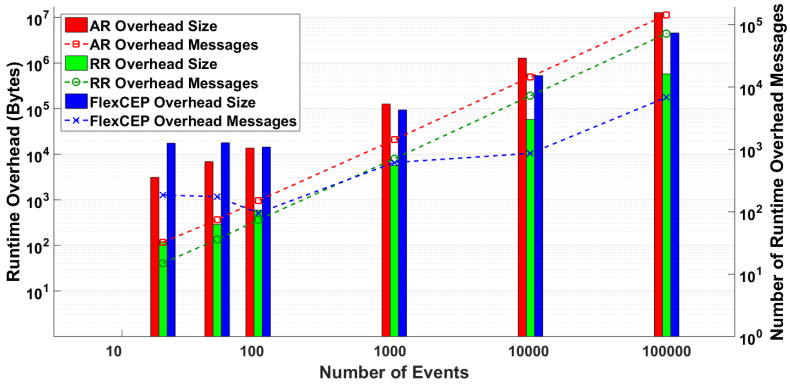
We first look into the communication overhead incurred by each of the approaches under consideration during failure-free and failure-ridden operation. In particular, we look at the runtime overhead and recovery overhead to understand how each of these approaches tackle abrupt failures at different points in time.



(a) Comparison of Runtime Overhead during Failure-Free Execution



(b) Comparison of Runtime Overhead with Failure of Downstream Node after 30% Input Events



(c) Comparison of Runtime Overhead with Failure of Downstream Node after 80% Input Events

Figure 28: Comparison of Runtime Overhead for Active Replication, Rollback Recovery, and FlexCEP under Different Failure Conditions in Quasi-Stationary Environments

Figure 28 shows the comparison between the different approaches with respect to the runtime overhead for different number of events transmitted. In AR, the runtime overhead is mainly dictated by the replicated set of events sent to the backup node in parallel to the events sent to the primary downstream node, as well as the acknowledgments sent by both nodes to the primary upstream node. The RR approach

only has runtime overhead coming from the acknowledgments sent to track event delivery. In our approach, runtime overhead is introduced by the beacons exchanged between nodes as well as the regular transfer of the operator state from the primary downstream node to the backup node.

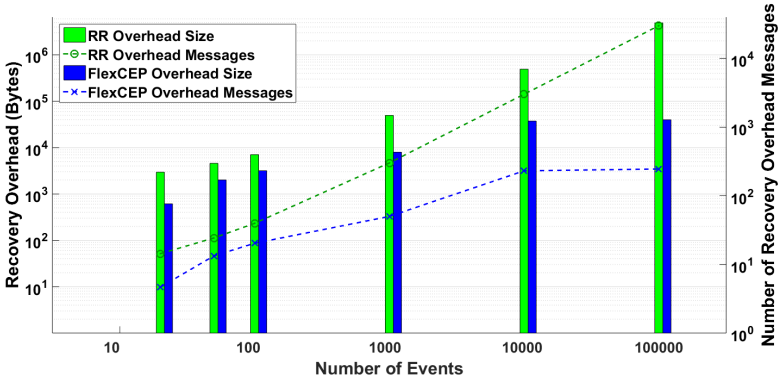
In Figure 28a, we observe the results for failure-free operation. Notice that both the axes are represented in the logarithmic scale. We observe that the runtime overhead incurred by AR surpasses both the other approaches, considerably, when the number of events transmitted—and therewith, the window size—increases. However, we see that FlexCEP incurs more runtime overhead for smaller number of events and smaller window sizes. This is mainly because of the size of the operator state and the frequency of state transfer from the primary downstream node to the backup. RR introduces a minimal runtime overhead for lower sets of events, but increases linearly with an exponential increase in the number of events. We notice that, while the number of messages exchanged using FlexCEP is below RR by a factor of 11.5 for the case with 100000 events, the total runtime overhead message size exceeds RR by a factor of 7.

Even during failure-ridden operation, we observe a similar pattern in the amount of runtime overhead incurred by each approach, as seen in Figures 28b and 28c. While the magnitude of the runtime overhead reduces for the case with an early failure, the reduction is proportional to the number of messages exchanged in total. We observe that FlexCEP incurs a relatively higher runtime overhead in the lower event regions, increasing to almost 10 times that of AR, compared to 8 during failure-free operation. Again, this is attributed to the size of the intermediate operator state transfer, which is done for every intermediate event with a window size of 2 (for the case with 20 events).

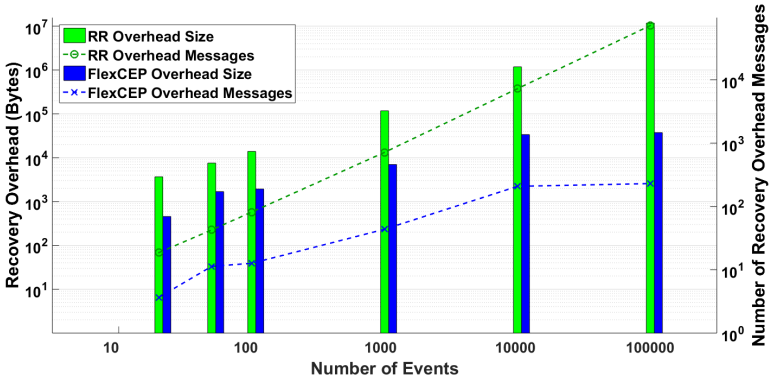
Communication Overhead: Recovery

Figure 29 presents the evaluation results with respect to the recovery overhead for each of the above approaches. Considering that AR requires minimal time to recover after failure (ideally zero), given that the second processing path already exists beforehand (via the backup node), we only evaluate the difference in recovery overhead between RR and FlexCEP. Again, notice the logarithmic scale of the axes in both figures.

We observe that the recovery overhead for rollback recovery is significantly higher than that of FlexCEP, in either case of failure. RR incurs a large recovery overhead due to the replay of buffered events to the backup node after failure is detected. The size of the buffer increases linearly with the number of events transmitted. On the other hand, FlexCEP integrates a regular trimming of the restore buffer $B_{restore}$ on the primary upstream node as and when events fill the intermediate buffer B_{int} on the primary downstream node (based on the acknowledgment rate, $R_{restore}$). As a result, a largely reduced number of events need to be transferred to the backup node for restoration purposes. We observe this more clearly in Figure 29b, where the gap between the recovery overhead for RR increases even further in comparison to that for FlexCEP, especially in the lower range of events. This is mainly due to the



(a) Comparison of Recovery Overhead with Failure of Downstream Node after 30% Input Events



(b) Comparison of Recovery Overhead with Failure of Downstream Node after 80% Input Events

Figure 29: Comparison of Recovery Overhead for Rollback Recovery and FlexCEP under Different Failure Conditions in Quasi-Stationary Environments

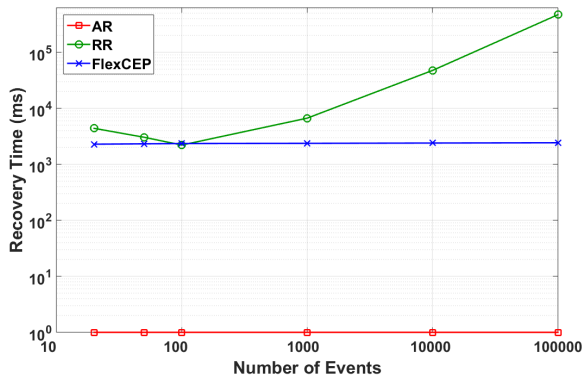
relatively constant number of events to be migrated in FlexCEP, despite the later occurrence of node failure.

Recovery Time

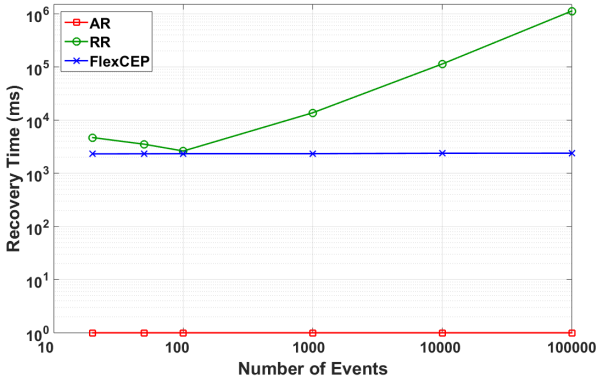
Having seen the differences between the three approaches in terms of communication overhead during failure-free and failure-ridden operation, we now look into the amount of time required by the approaches to recover after failure is detected. Note that the detection of failure is dependent on the rate of acknowledgments in RR and the rate of beacon transmission in FlexCEP.

Figure 30 shows the recovery time analysis with respect to different number of input events, as well as different failure time instants. Again, we use the logarithmic scale to accommodate the exponential growth in numbers. As mentioned earlier, AR incurs minimal delays in establishing the new path over the backup node, considering that the replicated path is already established beforehand.

In both Figures 30a and 30b, we observe that the recovery time for RR increases linearly with increase in the number of events, reaching a total of 470 seconds for



(a) Comparison of Recovery Time with Failure of Downstream Node after 30% Input Events



(b) Comparison of Recovery Time with Failure of Downstream Node after 80% Input Events

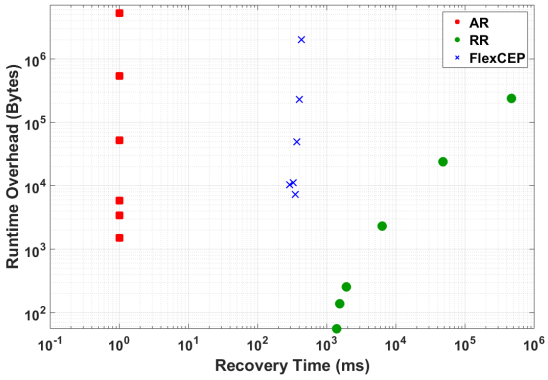
Figure 30: Comparison of Recovery Time for Rollback Recovery and FlexCEP under Different Failure Conditions in Quasi-Stationary Environments

100000 input events. Contrarily, recovery time using FlexCEP is relatively constant through all input event sizes ($\mu = 2.3s$). The main contributing factor towards recovery time in FlexCEP is the time taken to recognize the occurrence of a failure, based on the beaconing frequency R_{beacon}.

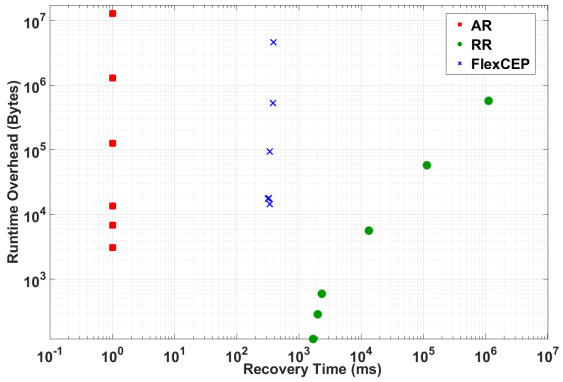
Runtime Overhead vs Recovery Time

Figure 31 shows the comparison of runtime overhead against recovery time for different input event sizes, similar to the figure shown in Section 6.1. In order to understand just the recovery time after failure has been detected, we subtracted the failure recognition time from the overall recovery times for FlexCEP and RR. We observe that RR exhibits an exponential approach curve, starting low for lower input events and increasing steadily with increase in events.

In both the failure cases, we notice that our approach FlexCEP moves between AR and RR, demonstrating a considerably shorter recovery time in the order of a few hundred milliseconds across all event sizes as well as a lower runtime overhead in comparison to AR. By varying the restore buffer synchronization factor $\phi_{restore}$, we can achieve different levels of recovery time and runtime overhead, depending on



(a) Comparison of Runtime Overhead against Recovery Time with Failure of Downstream Node after 30% Input Events



(b) Comparison of Runtime Overhead against Recovery Time with Failure of Downstream Node after 80% Input Events

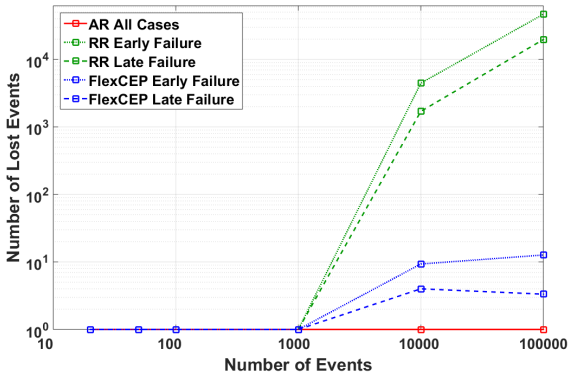
Figure 31: Comparison of Runtime Overhead against Recovery Time for All Approaches under Different Failure Conditions and Different Number of Events in Quasi-Stationary Environments

the needs of the application at hand. We discuss this trade-off setting in more detail in Section 6.5.

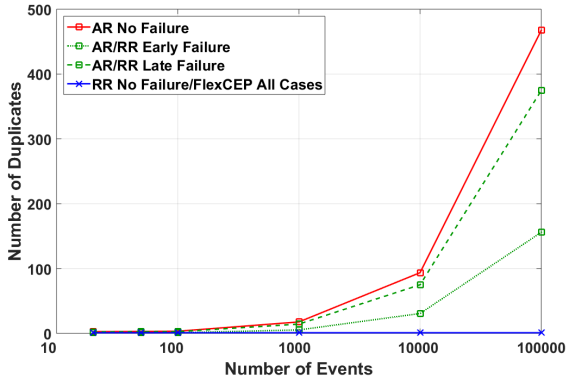
Faulty Events

Finally, we compare the performance of the three approaches in terms of event loss and duplicates, both of which we term together as *faulty* events. Recall from Section 6.1 that a loss of events can lead to increases in either false positives or false negatives, which can in turn have dire consequences for the end application. Furthermore, an increase in the number of duplicates requires the consumer node to introduce additional filter operations to sort out the valid entries from the invalid ones. Effectively, it is highly desirable for any reliability mechanism to incur minimal event loss and duplicates.

Figure 32 shows the results for event loss and duplicate count for increasing number of events. In Figure 32a, we observe that, given the nature of AR to always have at least one redundant path, the number of events lost is zero (shown as 1 in the log



(a) Comparison of Event Loss for Different Failure Conditions



(b) Comparison of Number of Duplicates for Different Failure Conditions

Figure 32: Comparison of Event Loss and Duplicates across All Approaches under Different Failure Conditions in Quasi-Stationary Environments

plot to make them visible). However, RR incurs an excessively high number of lost events when the number of events generated increases beyond 10000, which also indicates a higher R_{eg} of 100 events/s as per Table 16. This is mainly due to the fact that the primary upstream node has to be blocked during the recovery process, resulting in a loss of incoming events due to buffer overflow. Notice that the event loss is lower when the failure occurs later, considering that fewer events are lost during recovery blockage. The minimal number of lost events using FlexCEP is also caused due to the short period of time, where the primary upstream node is blocked in order to transmit the contents of $B_{restore}$ to the backup node. While the loss count is negligibly small compared to that for RR, it can be further reduced by increasing the restore buffer synchronization factor, $\phi_{restore}$.

The number of duplicates received at the consumer node for different event sizes is shown in Figure 32b. Note the linear scale used on the y-axis for the number of duplicates, still keeping the x-axis logarithmic. In AR, the number of duplicates is equal to the events processed by the backup node during failure-free runtime, and by the primary downstream node during failure-ridden runtime. Similarly, in RR, the events generated by the backup node after restore partially includes some events that were generated by the primary downstream node before its failure. In the case

of FlexCEP, given the transfer of operator state to the backup node, the processing operation can be continued exactly where the primary node left off. This property of FlexCEP effects zero duplicates at the consumer node (again, shown as 1 in the log plot).

6.4.2 *Performance Analysis in Mobile Environments*

In this section, we look into the performance results for the evaluation of the approaches FlexCEP and 1AR in mobile environments. As described in Section 6.3.3, we use the mobility trace datasets based on vehicular traffic on the Spanish highways around Madrid to simulate high-speed nodes. Note that we use different values for R_{eg} and the window size, as seen in Table 17. As per the mobility trace dataset, the beaconing rate R_{beacon} increases to 2 beacons/s. We set the *failValue* to 0.7, such that the primary upstream node starts migration to the backup node when this failure probability is reached with respect to the primary downstream node.

Communication Overhead

Just as we did in the quasi-stationary scenario, we first look into the communication overhead incurred by both the approaches under consideration. Considering that we have a highly mobile environment, we do not consider the scenario of failure-free operation during this analysis.

Figure 33 shows the results for the comparison between 1AR and FlexCEP with regard to the runtime overhead for increasing R_{eg}, and therefore an increasing window size from size 4 for a rate of 1 event/s and size 32 for 1000 events/s, respectively. 1AR mainly incurs a runtime overhead because of the beacons exchanged among all nodes as well as the set of events sent to the primary downstream node during the period between the detection of impending failure and subsequent failure of the primary downstream node. In FlexCEP, as before, the runtime messages are comprised of operator state transfer to the backup node, acknowledgments for restore buffer trimming, and beacon exchange.

Interestingly, we observe that the number of messages exchanged during normal runtime is relatively constant between the two approaches. However, we notice that the message overhead in bytes in FlexCEP increases steadily with the number of events transmitted, reaching a value of 473 kB for 1000 events/s, compared to just 75 kB for 1AR. This is mainly caused by the increased amount of operator state transfer in FlexCEP with increases in R_{eg}.

In Figure 34, we look at the recovery overhead in each of these two approaches when an impending failure of the primary downstream node has been detected (based on $\mathcal{P}_{fail} \geqslant failValue$). Just as in RR, 1AR needs to transfer all the output events stored in its buffer to the backup node to restore the current processing state on the backup node. Consequently, we observe that the recovery overhead for 1AR increases linearly with increases to R_{eg}, since this implies that more number of events are stored in its buffer for restoration purposes. Contrarily, FlexCEP incurs almost a constant recovery overhead across all values of R_{eg}, owing mainly to the constant

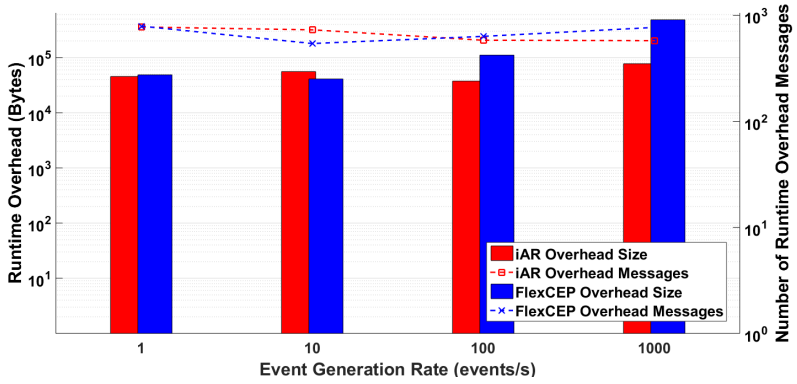


Figure 33: Comparison of Runtime Overhead for Improved Active Replication (iAR) and FlexCEP in Mobile Environments

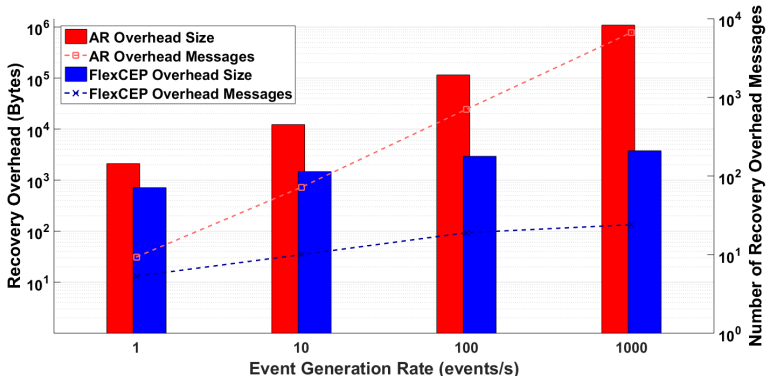


Figure 34: Comparison of Recovery Overhead for Improved Active Replication (iAR) and FlexCEP in Mobile Environments

trimming of the restore buffer $B_{restore}$ on the primary upstream node. This results in the stark differences in recovery overhead between FlexCEP and iAR, especially for the case with $R_{eg} = 1000$ events/s, where the recovery overhead of iAR is almost 300 times that of FlexCEP. In FlexCEP, the slight increase in the number of messages and the recovery overhead size is caused by the increase in window size, resulting in more events in $B_{restore}$. This can be combated by increasing the acknowledgment rate $R_{restore}$, trading off with increased runtime overhead.

Recovery Time

Figure 35 shows the results for the comparison between the two approaches with respect to recovery time, after an impending failure of the primary downstream node has been detected. Notice that we use a linear scale on the y-axis for the recovery time in this case. We can see that, much like RR, iAR suffers from increased recovery times for increases in R_{eg}. While the recovery time of iAR for $R_{eg} = 1$ event/s is relatively comparable to that of FlexCEP at 466 ms and 349 ms, respectively, it increases dramatically to 7.9 seconds and 8.2 seconds when R_{eg} increases to 100 events/s and 1000 events/s, respectively. This attributes to the increased number of event mes-

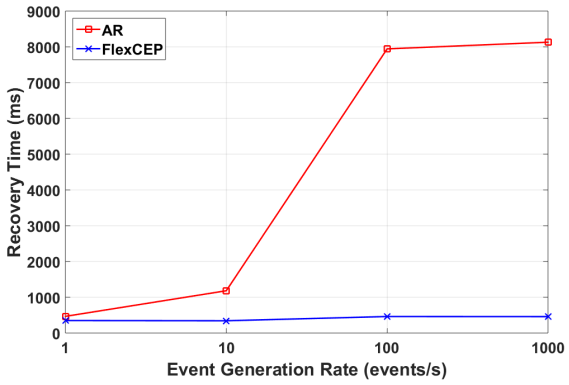


Figure 35: Comparison of Recovery Time for Improved Active Replication (iAR) and FlexCEP in Mobile Environments

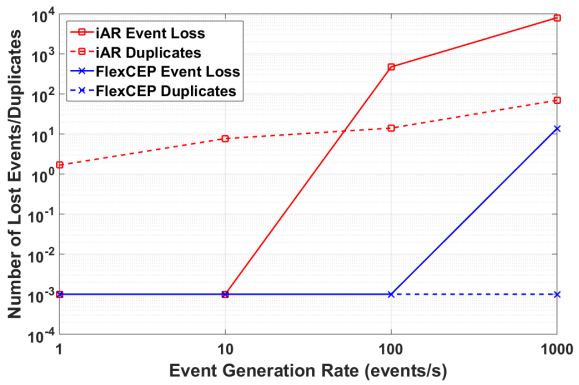


Figure 36: Comparison of Event Loss and Duplicates for Improved Active Replication (iAR) and FlexCEP

sages to be restored on the backup node upon failure detection, as seen in Figure 34. Just as we observed in Figure 30 for the quasi-stationary case, FlexCEP does not incur any major increases in recovery time with increases in R_{eg}, increasing to just 459 ms (from 349 ms) for the case with 1000 events/s.

Faulty Events

Finally, we look at the performance of the two approaches with respect to the number of *faulty* events—i.e., number of events lost and number of duplicate events—with increases in R_{eg}. Figure 36 shows the corresponding results, using a logarithmic scale for the number of lost events/duplicates. We have substituted the zero values in the number of lost events/duplicates with 0.001 in order to make them visible on a log plot. In terms of event loss, we observe that iAR incurs larger event losses when R_{eg} increases to 100 events/s and beyond. Just as seen in the quasi-stationary case, this is caused due to the blockage of the primary upstream node during the restoration phase. Furthermore, it can also be attributed to the inability of the backup node to process the larger surge of events caused by larger number of buffered events. Even FlexCEP suffers from increased loss of events for the case with $R_{eg} = 1000$ events/s,

resulting in an average event loss of 13 events, albeit negligible compared to around 8000 in 1AR.

With respect to the number of duplicates, FlexCEP again proves that appropriate operator state transfer can help in avoiding unnecessary additional duplicate filtering at the consumer node. In case of 1AR, we observe that the number of duplicates increases steadily with increases in R_{eg}, increasing from an average of 2 events for $R_{eg} = 1$ event/s to 70 events for 1000 events/s. This is mainly caused by the parallel processing of events during the period between failure detection and eventual failure of the primary downstream node.

6.5 DISCUSSION

In this chapter, we designed and evaluated a novel approach called FlexCEP for flexible execution of CEP operators in a dynamic D2D environment. We introduced an adaptive buffer management scheme, which incorporates fine-grained intermediate state of the CEP operators to reduce the amount of recovery overhead and recovery time incurred during operator migration. Furthermore, to reduce the amount of runtime overhead during failure-free operation, even in highly mobile environments, we introduced a mobility-aware predictive mechanism that incorporates the exchange of beacons to estimate the failure probability of the other nodes in the system.

Through our evaluations in both a quasi-stationary environment as well as a highly mobile vehicular environment, we showed that we achieve lower runtime overhead compared to active replication and significantly lower recovery overhead and recovery time compared to rollback recovery, especially at larger rates of event generation. Considering that FlexCEP incorporates the transfer of fine-grained and up-to-date operator state information, it incurs near-zero event loss and event duplication.

In our evaluations, we considered a case where the restore buffer $B_{restore}$ on the primary upstream node is updated (and trimmed) in accordance with the restore buffer synchronization factor, $\phi_{restore} = 2/\texttt{WINDOW\_SIZE}$. By varying the value of $\phi_{restore}$ between 1 and `WINDOW_SIZE`, we can achieve different degrees of runtime and recovery overhead, thus allowing for an adaptive approach for different application settings. For example, with $\phi_{restore} = 1/\texttt{WINDOW\_SIZE}$, FlexCEP allows us to choose those opportune moments when the state of the operator is effectively null, which roughly corresponds to the *savepoint* approach developed by Koldehofe et al. [116]. A larger value of $\phi_{restore}$ reduces the runtime overhead considerably (by 2 in our case), but can potentially increase the recovery overhead significantly. Similarly, setting $\phi_{restore} = 1$ (i.e., trimming the restore buffer for every event that arrives in the intermediate buffer) will result in a considerably higher runtime overhead (comparable with that of active replication), but lead to much lower recovery overhead and time, given that the size of $B_{restore}$ will be at the optimal minimum.

While our evaluations showcase the advantages of FlexCEP over active replication and rollback recovery, they do not consider the possible consequences of network failure and real-world connectivity. Possible loss of connectivity or loss of events due to the wireless infrastructure can be detrimental factors to the functioning of a

distributed CEP system. In our work, we mainly focused on analyzing the option to adapt operator migration in dynamic environments in accordance with the operator state and the node characteristics. The evaluation of one such approach in real-world CEP systems, with possibly multiple node failures, is an open research question.

Finally, FLEXCEP can also be used to migrate CEP operators from (seemingly) less-trustworthy nodes to more-trustworthy nodes, if the trust relations of the users change during the execution of an operator graph. In our work, TRUSTCEP first analyzes the trust recommendations and then uses the established trust levels for the placement and execution of CEP operators. However, it can be modified to incorporate trust recommendations on the fly, based on the activity of other users. Consequently, the migration of the operators can be done in accordance with the current trust values (which are also exchanged using trust recommendations). While we indicate this connection between the TRUSTCEP and FLEXCEP components for now, the evaluation of their combined functionality is subject to future work.

7 CONCLUSION

We now conclude this thesis by summarizing the main contributions of our work. We discuss the results obtained from the evaluation of the three components of our work and draw conclusions on their applicability in the future. In doing so, we also discuss the open issues and provide an outlook towards future work.

7.1 SUMMARY OF THE THESIS

In our work, the main goal was the development of a distributed CEP system that allows for privacy-aware and reliable processing of context data in dynamic D2D environments as part of the IoT. In Chapter 1, we motivated the usage of CEP for context processing in the IoT, and provided an extensive overview of the fundamentals behind CEP in Chapter 2. Proceeding from the basics of CEP, we motivated the main challenges in Chapter 1—differing privacy constraints, presence of adversaries, and dynamic changes in D2D environments—that must be overcome by one such distributed system.

Based on the system requirements derived from the above challenges, we identified event processing, trusted computing, and human relationship analysis as the three main fields for our work. We comprehensively analyzed existing state of the art in the above fields in Chapter 2, focusing on prevalent approaches towards privacy-awareness and reliability. In turn, as our overall contribution, we proposed a distributed CEP system that incorporates trust-based and flexible execution of CEP operators in dynamic D2D environments; Chapter 3 presents an overview of the proposed system. In the following, we shall summarize the key aspects of the three main contributions of our work, along with a discussion of the corresponding evaluation results.

Contribution 1: Estimating User Relationships to Identify Their Privacy Constraints

Our first contribution deals with the automatic estimation of user relationships in order to identify the privacy constraints of the users. We leverage sociological findings that the type (social circle) and strength (tie strength) of a human relationship have a direct correspondence with the trust in the relationship, and therefore, with the amount of information shared and disclosed between the two users. In turn, we present a method to estimate the social circle and tie strength based on the communication characteristics between the users, presented in Chapter 4.

Based on our evaluations using supervised machine learning algorithms, we obtained an average accuracy of 77.36% ($\kappa = 0.33$) in estimating the social circles, and 75.81% ($\kappa = 0.42$) in identifying the strong relationships. We observed that *family* and *work* contacts exhibit typical temporal communication patterns. We also observed

certain interesting characteristics—e.g., the number of communication channels per contact and the number of emoticons used in messages are positive indicators of a stronger relationship—allowing us to establish diverse interaction patterns across different relationships.

Contribution 2: Trust-Based Execution of Distributed CEP Systems

The results from our FamApp analysis provided the basis for our next contribution called TrustCEP, explained in Chapter 5. As part of this contribution, we proposed a trust-based approach for distributed CEP, by facilitating a privacy-aware operator placement and execution in D2D environments. Leveraging the concept of *behavioral trust* that formulates trust based on user behavioral patterns (e.g., communication patterns), we propose a trust management model, TrustCEP. TrustCEP evaluates the direct trust between users based on observations in their communication patterns and interaction history, using the key influential features established in Chapter 4. Furthermore, TrustCEP accounts for the evolution of trust by incorporating a robust scheme for trust recommendations, which also allows us to identify the adversaries in the system. The derived trust relations between the users allow us to deploy a privacy-aware placement algorithm to place and execute the operators on the devices that are trustworthy.

Based on analytical evaluation tests, we found that our approach prevents adversary attacks as long as the adversaries are not in the majority. We also observed that our approach allows for the detection of adversaries in the majority, when the adversaries fail to infiltrate the entire network. We also conducted a performance analysis of our approach by deploying it in a smartphone-based distributed CEP system, which allows for direct interaction and collaborative processing. We observed that our approach inflicts only a marginally higher battery consumption (2-6%) on contemporary smartphones, in comparison to other privacy-negligent approaches.

Contribution 3: Reliable Execution of Distributed CEP Systems in Dynamic Environments

Finally, our third contribution deals with the reliable execution of CEP operators in a dynamic D2D environment, focusing on the resource-constrained and mobile nature of D2D-based networks in the IoT. To facilitate a reliable execution of distributed CEP, we developed a flexible operator migration approach called FlexCEP, as described in Chapter 6. Through FlexCEP, we devised a fine-grained and proactive approach to migrate CEP operator state from one node to another, depending on the mobility as well as the prevailing internal conditions of the nodes in the network. In particular, we propose an adaptive buffer management approach to allow for appropriate rollback recovery in case of failures. Furthermore, we propose a predictive analysis of the availability of the nodes by keeping track of vital information about the nodes' instantaneous characteristics—e.g., location coordinates or energy levels.

In our evaluations, we compared our approach against active replication and rollback recovery approaches in both quasi-stationary as well as highly mobile environments. Our evaluation results show that we achieve significantly lower communica-

tion overhead (runtime and recovery) and lower recovery times compared to both approaches towards reliability in distributed CEP systems. FlexCEP also results in near-zero event loss and duplicates, owing mainly to the fine-grained nature of operator state transfer for recovery. By varying the rate at which the nodes trim their buffer content, we can achieve a wider spectrum of communication overhead and recovery time, depending on the requirements of the application at hand.

7.1.1 *Conclusions*

With the advent of user-centric applications based on situational context in the IoT, the need for preserving user privacy becomes increasingly important. Furthermore, the dynamic nature of the IoT introduces additional challenges to the proper working of these applications, especially given the distributed nature of information sources and interested consumers in the IoT. The combination of the three main contributions of this thesis allows for the realization of a privacy-aware and reliable CEP system for dynamic D2D environments, as described in Section 3.3.

In our work, we maintain the users as the central stakeholders of the system. Consequently, our approach for a privacy-aware and reliable CEP system incorporates the privacy constraints of the users as well as the resource-constrained and mobile nature of their devices. By incorporating the trust relations between the users during the placement and execution of the CEP operators, we allow for event transfer only to those devices that are trustworthy to the user(s) at hand. We model the trust relations based on the behavioral aspects of user relationships, thus avoiding burdensome methods that require the attention of the users [142, 170]. Finally, by accounting for their mobility and device resources, we provide for an adaptive and flexible approach to facilitate reliability in dynamic D2D environments.

7.2 OUTLOOK

The results of this thesis can be used in many application scenarios pertinent to the IoT. The three scenarios described in Section 3.1—and used as the basis for the evaluations—are inherent aspects of context-aware applications for traffic monitoring, home automation, healthcare, and environmental planning, among others, where the sensor data from the environment are processed to obtain higher-level information about the situational user context. While this thesis only concentrates on D2D environments, with short-range communication between devices, the transfer of these approaches to the global scale is subject to future work. *Fog computing* [35, 214] is an upcoming network paradigm that brings cloud computing to the edge of the network, and therefore allows for low-latency and mobility-aware services, especially in the context of the IoT [34, 35]. Another bright prospect for a new communication paradigm is *Named-Data Networking* (NDN) [105], which deals with the direct retrieval of information based on its name, instead of the host-based packet delivery prevalent in the Internet nowadays. The combination of NDN and fog computing is a very promising field for future research—as shown in the initial work by

Amadeo et al. [8]—where the approaches proposed in this thesis provide the initial groundwork for privacy-aware and reliable communication.

An important aspect of our work revolves around the trust relations between users. While our work incorporates one dimension of trust, in reality, human relationships exhibit different trust levels in different contexts, e.g., depending on the locative and temporal conditions. Many articles in related literature have focused on understanding the nuances of sharing patterns based on human relationships [27, 51, 114, 153, 170], and pointed out the significance of contextual information in governing the sharing patterns. Incorporating these factors into the trust measurement framework and dealing with trust recommendations in multiple contexts present interesting directions for future work in trusted distributed computing in the IoT.

A correlated aspect to trust is security in distributed D2D environments. While capturing the trust relations allows us to incorporate privacy-awareness in the proposed system, the ultimate implementation of the system necessitates strong underlying security mechanisms. Recently, Intel® released a new technology called *Software Guard eXtensions* (SGX) to allow application developers to shield selected parts of their code from any external modifications using secure *enclaves* [102]. In turn, Intel SGX ensure the confidentiality and integrity of the data within enclaves [11]. However, while SGX are currently available for Intel® motherboards, there are no feasible hardware mechanisms available for user smartphones at the moment. Another promising mechanism is homomorphic encryption, which allows for the processing of encrypted data without revealing the contents of the data themselves [162]. While current implementations of homomorphic encryption are either insufficient for IoT-based applications or unfeasible for resource-constrained devices such as user smartphones [195], a possible future research direction is the development of such encryption mechanisms for the IoT.

Concerning reliability in distributed CEP, in our work, we proposed a flexible approach that allows for different trade-offs in terms of communication overhead and recovery time, depending on the needs of the application at hand. An interesting direction for future research is the development of transition mechanisms that adapt the underlying algorithm to the environmental conditions. For example, when the node mobility is low, the amount of runtime communication overhead can be reduced, considering that the probability of node failure is lower. Conversely, in highly-mobile environments, the runtime overhead can be traded off for reduced recovery overhead and recovery time. The Collaborative Research Centre "MAKI" is currently working on transitions in event-based communication systems, and especially on the proactive planning and coordination of transitions in such systems [37]. Adapting the *functionality* of a distributed CEP system to varying environmental conditions is a compelling aspect for future work.

ACKNOWLEDGMENTS

This work has been [co-]funded by the Social Link Project within the Loewe Program for Excellence in Research, Hesse, Germany.

BIBLIOGRAPHY

[1] Daniel J Abadi, Yanif Ahmad, Magdalena Balazinska, Ugur Cetintemel, Mitch Cherniack, Jeong-Hyon Hwang, Wolfgang Lindner, Anurag Maskey, Alex Rasin, Esther Ryvkina, Nesime Tatbul, Ying Xing, and Stan Zdonik. "The Design of the Borealis Stream Processing Engine." In: *Proceedings of the 2005 Conference on Innovative Data Systems Research (CIDR)*. Asilomar, CA, USA, 2005, pp. 277–289.

[2] Karl Aberer and Zoran Despotovic. "Managing Trust in a Peer-2-Peer Information System." In: *Proceedings of the Tenth International Conference on Information and Knowledge Management (CIKM)*. Atlanta, GA, USA: ACM, 2001, pp. 310–317.

[3] Mehran Abolhasan, Tadeusz Wysocki, and Eryk Dutkiewicz. "A Review of Routing Protocols for Mobile Ad Hoc Networks." In: *Ad Hoc Networks* 2.1 (2004), pp. 1–22.

[4] Dominic Abrams and Michael A Hogg. "Metatheory: Lessons from Social Identity Research." In: *Personality and Social Psychology Review* 8.2 (2004), pp. 98–106.

[5] Sibel Adali, Robert Escriva, Mark K Goldberg, Mykola Hayvanovych, Malik Magdon-ismail, Boleslaw K Szymanski, William A Wallace, and Gregory Williams. "Measuring Behavioral Trust in Social Networks." In: *IEEE International Conference on Intelligence and Security Informatics (ISI)*. Vancouver, BC, Canada: IEEE, 2010, pp. 150–152.

[6] Carlisle Adams and Steve Lloyd. *Understanding PKI: Concepts, Standards, and Deployment Considerations*. Addison-Wesley Professional, 2003.

[7] Pedro Alves and Paulo Ferreira. "Radiator: Context Propagation Based on Delayed Aggregation." In: *Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW)*. San Antonio, TX, USA: ACM, 2013, pp. 249–260.

[8] Marica Amadeo, Claudia Campolo, and Antonella Molinaro. "NDNe: Enhancing Named Data Networking to Support Cloudification at the Edge." In: *IEEE Communications Letters* 20.11 (2016), pp. 2264–2267.

[9] Lisa Amini, Henrique Andrade, Ranjita Bhagwan, Frank Eskesen, Richard King, Philippe Selo, Yoonho Park, and Chitra Venkatramani. "SPC: A Distributed, Scalable Platform for Data Mining." In: *Proceedings of the 4th International Workshop on Data Mining Standards, Services and Platforms*. ACM. Philadelphia, PA, USA, 2006, pp. 27–37.

[10] Reid Andersen, Christian Borgs, Jennifer Chayes, Uriel Feige, Abraham Flaxman, Adam Kalai, Vahab Mirrokni, and Moshe Tennenholtz. "Trust-based Recommendation Systems: An Axiomatic Approach." In: *Proceedings of the 17th International Conference on World Wide Web (WWW)*. Beijing, China: ACM, 2008, pp. 199–208.

[11] Sergei Arnautov, Bohdan Trach, Franz Gregor, Thomas Knauth, Andre Martin, Christian Priebe, Joshua Lind, Divya Muthukumaran, Dan O'Keeffe, Mark Stillwell, et al. "SCONE: Secure Linux Containers with Intel SGX." In: *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. Savannah, GA, USA: USENIX Association, 2016, pp. 689–703.

[12] Daniel Avrahami and Scott E. Hudson. "Communication Characteristics of Instant Messaging: Effects and Predictions of Interpersonal Relationships." In: *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work (CSCW)*. Banff, Alberta, Canada: ACM, 2006, pp. 505–514.

[13] Baruch Awerbuch, David Holmer, Cristina Nita-Rotaru, and Herbert Rubens. "An On-Demand Secure Routing Protocol Resilient to Byzantine Failures." In: *Proceedings of the 1st ACM Workshop on Wireless Security*. ACM. Atlanta, GA, USA, 2002, pp. 21–30.

[14] Brian Babcock, Shivnath Babu, Mayur Datar, Rajeev Motwani, and Jennifer Widom. "Models and Issues in Data Stream Systems." In: *Proceedings of the Twenty-first ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*. Madison, WI, USA: ACM, 2002, pp. 1–16.

[15] Shivnath Babu and Jennifer Widom. "Continuous Queries over Data Streams." In: *ACM Sigmod Record* 30.3 (2001), pp. 109–120.

[16] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. "SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining." In: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC)*. Vol. 10. Valletta, Malta, 2010, pp. 2200–2204.

[17] Lars Backstrom and Jon Kleinberg. "Romantic Partnerships and the Dispersion of Social Ties: A Network Analysis of Relationship Status on Facebook." In: *Proceedings of the 2014 Conference on Computer Supported Cooperative Work (CSCW)*. Baltimore, MD, USA: ACM, 2014, pp. 831–841.

[18] Jean Bacon, David Eyers, Ken Moody, and Lauri Pesonen. "Securing Publish/-Subscribe for Multi-Domain Systems." In: *Proceedings of the ACM/IFIP/USENIX 2005 International Conference on Middleware*. Springer-Verlag New York, Inc. Grenoble, France, 2005, pp. 1–20.

[19] Jean Bacon, David Eyers, Thomas FJ-M Pasquier, Jatinder Singh, Ioannis Papagiannis, and Peter Pietzuch. "Information Flow Control for Secure Cloud Computing." In: *IEEE Transactions on Network and Service Management* 11.1 (2014), pp. 76–89.

[20] Brian P Bailey and Joseph A Konstan. "On the Need for Attention-aware Systems: Measuring Effects of Interruption on Task Performance, Error Rate, and Affective State." In: *Computers in Human Behavior* 22.4 (2006).

[21] Brian P. Bailey, Joseph A. Konstan, and John V. Carlis. "The Effects of Interruptions on Task Performance, Annoyance, and Anxiety in the User Interface." In: *IFIP TC13 International Conference on Human-Computer Interaction (INTERACT)*. Tokyo, Japan: IOS Press, 2001, pp. 593–601.

[22] Magdalena Balazinska, Hari Balakrishnan, Samuel R Madden, and Michael Stonebraker. "Fault-Tolerance in the Borealis Distributed Stream Processing System." In: *ACM Transactions on Database Systems (TODS)* 33.1 (2008), p. 3.

[23] Magdalena Balazinska, Hari Balakrishnan, and Michael Stonebraker. "Contract-Based Load Management in Federated Distributed Systems." In: *First Symposium on Networked Systems Design and Implementation (NSDI)*. Vol. 4. San Francisco, CA, USA: USENIX Association, 2004, pp. 15–15.

[24] Xuan Bao, Jun Yang, Zhixian Yan, Lu Luo, Yifei Jiang, Emmanuel Munguia Tapia, and Evan Welbourne. "CommSense: Identify Social Relationship with Phone Contacts via Mining Communications." In: *2015 16th IEEE International Conference on Mobile Data Management (MDM)*. Vol. 1. IEEE. Pittsburgh, PA, USA, 2015, pp. 227–234.

[25] John S Baras and Tao Jiang. "Cooperation, Trust and Games in Wireless Networks." In: *Advances in Control, Communication Networks, and Transportation Systems* (2005), pp. 183–202.

[26] Raphaël Barazzutti, Pascal Felber, Hugues Mercier, Emanuel Onica, and Etienne Rivière. "Thrifty Privacy: Efficient Support for Privacy-preserving Publish/Subscribe." In: *Proceedings of the 6th ACM International Conference on Distributed Event-Based Systems (DEBS)*. Berlin, Germany: ACM, 2012, pp. 225–236.

[27] Louise Barkhuus. "The Measurement of Privacy: Using Contextual Integrity to Reconsider Privacy in HCI." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. Austin, TX, USA: ACM, 2012, pp. 367–376.

[28] Lars Baumgärtner, Christian Strack, Bastian Hoßbach, Marc Seidemann, Bernhard Seeger, and Bernd Freisleben. "Complex Event Processing for Reactive Security Monitoring in Virtualized Computer Systems." In: *Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems (DEBS)*. Oslo, Norway: ACM, 2015, pp. 22–33.

[29] Stephen Bell, Alisdair McDiarmid, and James Irvine. "Nodobo: Mobile Phone as a Software Sensor for Social Network Research." In: *IEEE 73rd Vehicular Technology Conference (VTC)*. Yokohoma, Japan: IEEE, 2011, pp. 1–5.

[30] Paolo Bellavista, Antonio Corradi, Mario Fanelli, and Luca Foschini. "A Survey of Context Data Distribution for Mobile Ubiquitous Systems." In: *ACM Computing Surveys (CSUR)* 44.4 (2012), p. 24.

[31] Oladayo Bello and Sherali Zeadally. "Intelligent Device-to-Device Communication in the Internet of Things." In: *IEEE Systems Journal* 10.3 (2016), pp. 1172–1182.

[32] András Belokosztolszki, David M Eyers, Peter R Pietzuch, Jean Bacon, and Ken Moody. "Role-Based Access Control for Publish/Subscribe Middleware Architectures." In: *Proceedings of the 2nd International Workshop on Distributed Event-Based Systems*. ACM. San Diego, CA, USA, 2003, pp. 1–8.

[33] Claudio Bettini, Oliver Brdiczka, Karen Henricksen, Jadwiga Indulska, Daniela Nicklas, Anand Ranganathan, and Daniele Riboni. "A Survey of Context Modelling and Reasoning Techniques." In: *Pervasive and Mobile Computing* 6.2 (2010), pp. 161–180.

[34] Flavio Bonomi, Rodolfo Milito, Preethi Natarajan, and Jiang Zhu. "Fog Computing: A Platform for Internet of Things and Analytics." In: *Big Data and Internet of Things: A Roadmap for Smart Environments*. Springer, 2014, pp. 169–186.

[35] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. "Fog Computing and Its Role in the Internet of Things." In: *Proceedings of the First Edition of the Workshop on Mobile Cloud Computing (MCC)*. ACM. Helsinki, Finland, 2012, pp. 13–16.

[36] Alejandro Buchmann and Boris Koldehofe. "Complex Event Processing." In: *IT-Information Technology Methoden und innovative Anwendungen der Informatik und Informationstechnik* 51.5 (2009), pp. 241–242.

[37] *C2: Information-Centered Perspective - Event-Based Communication in Programmable Networks*. URL: `https://www.maki.tu-darmstadt.de/sfb_maki/teilprojekte/projektbereich_c/tp_c02/tp_c2.en.jsp`.

[38] Valeria Cardellini, Vincenzo Grassi, Francesco Lo Presti, and Matteo Nardelli. "Optimal Operator Placement for Distributed Stream Processing Applications." In: *Proceedings of the 10th ACM International Conference on Distributed and Event-Based Systems*. Irvine, CA, USA: ACM, 2016, pp. 69–80.

[39] Antonio Carzaniga, David S Rosenblum, and Alexander L Wolf. "Achieving Scalability and Expressiveness in an Internet-scale Event Notification Service." In: *Proceedings of the Nineteenth Annual ACM Symposium on Principles of Distributed Computing*. ACM. Portland, OR, USA, 2000, pp. 219–227.

[40] Antonio Carzaniga, David S Rosenblum, and Alexander L Wolf. "Design and Evaluation of a Wide-Area Event Notification Service." In: *ACM Transactions on Computer Systems (TOCS)* 19.3 (2001), pp. 332–383.

[41] Antonio Carzaniga and Alexander L Wolf. "Content-Based Networking: A New Communication Infrastructure." In: *Workshop on Infrastruture for Mobile and Wireless Systems*. Springer. Scottsdale, AZ, USA, 2001, pp. 59–68.

[42] Raul Castro Fernandez, Matteo Migliavacca, Evangelia Kalyvianaki, and Peter Pietzuch. "Integrating Scale-Out and Fault-Tolerance in Stream Processing using Operator State Management." In: *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*. New York, NY, USA: ACM, 2013, pp. 725–736.

[43] Younghun Chae, Lisa Cingiser DiPippo, and Yan Lindsay Sun. "Trust Management for Defending On-off Attacks." In: *IEEE Transactions on Parallel and Distributed Systems* 26.4 (2015), pp. 1178–1191.

[44] Sharma Chakravarthy and Deepak Mishra. "Snoop: An Expressive Event Specification Language for Active Databases." In: *Data & Knowledge Engineering* 14.1 (1994), pp. 1–26.

[45] K. Mani Chandy, Michel. Charpentier, and Agostino Capponi. "Towards a Theory of Events." In: *Proceedings of the 2007 Inaugural International Conference on Distributed Event-Based Systems (DEBS)*. Toronto, ON, Canada: ACM, 2007, pp. 180–187.

[46] K. Chandy and W. Schulte. *Event Processing: Designing IT Systems for Agile Companies*. McGraw-Hill, Inc., 2010.

[47] Ray Chen, Fenye Bao, MoonJeong Chang, and Jin-Hee Cho. "Dynamic Trust Management for Delay Tolerant Networks and Its Application to Secure Routing." In: *IEEE Transactions on Parallel and Distributed Systems* 25.5 (2014), pp. 1200–1210.

[48] Yuan Cheng, Jaehong Park, and Ravi Sandhu. "A User-to-User Relationship-Based Access Control Model for Online Social Networks." In: *IFIP Annual Conference on Data and Applications Security and Privacy (DBSec)*. Springer. Paris, France, 2012, pp. 8–24.

[49] Bruce Christianson and William S. Harbison. "Why isn't Trust Transitive?" In: *International Workshop on Security Protocols*. Springer. Cambridge, UK, 1996, pp. 171–176.

[50] Karen Church and Rodrigo de Oliveira. "What's Up with Whatsapp?: Comparing Mobile Instant Messaging Behaviors with Traditional SMS." In: *Proceedings of the 15th International Conference on Human-computer Interaction with Mobile Devices and Services (MobileHCI)*. Munich, Germany: ACM, 2013, pp. 352–361.

[51] Sunny Consolvo, Ian E Smith, Tara Matthews, Anthony LaMarca, Jason Tabert, and Pauline Powledge. "Location Disclosure to Social Relations: Why, When, & What People Want to Share." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. Portland, OR, USA: ACM, 2005, pp. 81–90.

[52] Gianpaolo Cugola and Alessandro Margara. "Complex Event Processing with T-REX." In: *Journal of Systems and Software* 85.8 (2012), pp. 1709–1728.

[53] Gianpaolo Cugola and Alessandro Margara. "Processing Flows of Information: From Data Stream to Complex Event Processing." In: *ACM Computing Surveys (CSUR)* 44.3 (2012), p. 15.

[54] Gianpaolo Cugola and Alessandro Margara. "Deployment Strategies for Distributed Complex Event Processing." In: *Computing* 95.2 (2013), pp. 129–156.

[55] Anupam Das and Mohammad Mahfuzul Islam. "SecuredTrust: A Dynamic Trust Computation Model for Secured Communication in Multiagent Systems." In: *IEEE Transactions on Dependable and Secure Computing* 9.2 (2012), pp. 261–274.

[56] Mieso K Denko, Tao Sun, and Isaac Woungang. "Trust Management in Ubiquitous Computing: A Bayesian Approach." In: *Computer Communications* 34.3 (2011), pp. 398–406.

[57] Prabal Dutta, Paul M Aoki, Neil Kumar, Alan Mainwaring, Chris Myers, Wesley Willett, and Allison Woodruff. "Common Sense: Participatory Urban Sensing Using a Network of Handheld Air Quality Monitors." In: *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys)*. ACM. Berkeley, CA, USA, 2009, pp. 349–350.

[58] Rahul Dwarakanath, Jérôme Charrier, Frank Englert, Ronny Hans, Dominik Stingl, and Ralf Steinmetz. "Analyzing the Influence of Instant Messaging on User Relationship Estimation." In: *2016 IEEE International Conference on Mobile Services (MS)*. San Francisco, CA, USA: IEEE, 2016, pp. 49–56.

[59] Rahul Dwarakanath, Boris Koldehofe, Yashas Bharadwaj, The An Binh Nguyen, David Eyers, and Ralf Steinmetz. "TrustCEP: Adopting a Trust-Based Approach for Distributed Complex Event Processing." In: *2017 IEEE International Conference on Mobile Data Management (MDM)*. Daejeon, South Korea: IEEE, 2017, pp. 30–39.

[60] Rahul Dwarakanath, Boris Koldehofe, and Ralf Steinmetz. "Operator Migration for Distributed Complex Event Processing in Device-to-Device Based Networks." In: *Proceedings of the 3rd Workshop on Middleware for Context-Aware Applications in the IoT*. Trento, Italy: ACM, 2016, pp. 13–18.

[61] Rahul Dwarakanath, Dominik Stingl, and Ralf Steinmetz. "Improving Inter-user Communication: A Technical Survey on Context-aware Communication." In: *PIK-Praxis der Informationsverarbeitung und Kommunikation* 38.1-2 (2015).

[62] Nathan Eagle and Alex Sandy Pentland. "Reality Mining: Sensing Complex Social Systems." In: *Personal and Ubiquitous Computing* 10.4 (2006), pp. 255–268.

[63] Nathan Eagle, Alex Sandy Pentland, and David Lazer. "Inferring Friendship Network Structure by Using Mobile Phone Data." In: *Proceedings of the National Academy of Sciences* 106.36 (2009), pp. 15274–15278.

[64] Neda Ebrahim-Khanjari, Wallace Hopp, and Seyed MR Iravani. "Trust and Information Sharing in Supply Chains." In: *Production and Operations Management* 21.3 (2012), pp. 444–464.

[65] Elmootazbellah Nabil Elnozahy, Lorenzo Alvisi, Yi-Min Wang, and David B Johnson. "A Survey of Rollback-Recovery Protocols in Message-Passing Systems." In: *ACM Computing Surveys (CSUR)* 34.3 (2002), pp. 375–408.

[66] eMarketer. *Number of smartphone users worldwide from 2014 to 2020 (in billions)*. 2015. URL: https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/.

[67] William Enck, Peter Gilbert, Seungyeop Han, Vasant Tendulkar, Byung-Gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick McDaniel, and Anmol N. Sheth. "TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones." In: *ACM Transactions on Computer Systems (TOCS)* 32.2 (2014), 5:1–5:29.

[68] Christian Esposito and Mario Ciampi. "On Security in Publish/Subscribe Services: A Survey." In: *IEEE Communications Surveys & Tutorials* 17.2 (2015), pp. 966–997.

[69] Opher Etzion, Peter Niblett, and David C Luckham. *Event Processing in Action*. Manning Greenwich, 2011.

[70] Patrick Eugster, Pascal A Felber, Rachid Guerraoui, and Anne-Marie Kermarrec. "The Many Faces of Publish/Subscribe." In: *ACM Computing Surveys (CSUR)* 35.2 (2003), pp. 114–131.

[71] Lujun Fang and Kristen LeFevre. "Privacy Wizards for Social Networking Sites." In: *Proceedings of the 19th International Conference on World Wide Web (WWW)*. Raleigh, NC, USA: ACM, 2010, pp. 351–360.

[72] Shelly D Farnham and Elizabeth F Churchill. "Faceted Identity, Faceted Lives: Social and Technical Issues with Being Yourself Online." In: *Proceedings of the 2011 Conference on Computer Supported Cooperative Work (CSCW)*. Hangzhou, China: ACM, 2011, pp. 359–368.

[73] Ludger Fiege, Andreas Zeidler, Alejandro Buchmann, Roger Kilian-Kehr, and Gero Mühl. "Security Aspects in Publish/Subscribe Systems." In: *Third International Workshop on Distributed Event-Based Systems (DEBS)*. IET. Edinburgh, Scotland, UK, 2004, pp. 44–49.

[74] Roy Friedman, Alex Kogan, and Yevgeny Krivolapov. "On Power and Throughput Tradeoffs of WiFi and Bluetooth in Smartphones." In: *IEEE Transactions on Mobile Computing* 12.7 (2013), pp. 1363–1376.

[75] Diego Gambetta. "Can We Trust Trust." In: *Trust: Making and Breaking Cooperative Relations* 13 (2000), pp. 213–237.

[76] Raghu K Ganti, Fan Ye, and Hui Lei. "Mobile Crowdsensing: Current State and Future Challenges." In: *IEEE Communications Magazine* 49.11 (2011).

[77] Mario Gerla, Eun-Kyu Lee, Giovanni Pau, and Uichin Lee. "Internet of Vehicles: From Intelligent Grid to Autonomous Cars and Vehicular Clouds." In: *2014 IEEE World Forum on Internet of Things (WF-IoT)*. IEEE. Seoul, South Korea, 2014, pp. 241–246.

[78] Eric Gilbert and Karrie Karahalios. "Predicting Tie Strength with Social Media." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. Boston, MA, USA: ACM, 2009, pp. 211–220.

[79] Javier Gozálvez, Miguel Sepulcre, and Ramon Bauza. "IEEE 802.11p Vehicle to Infrastructure Communications in Urban Environments." In: *IEEE Communications Magazine* 50.5 (2012), pp. 176–183.

[80] Marco Gramaglia, Oscar Trullols-Cruces, Diala Naboulsi, Marco Fiore, and Maria Calderon. "Vehicular Networks on Two Madrid Highways." In: *2014 Eleventh Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE. Singapore, Singapore, 2014, pp. 423–431.

[81] Mark Granovetter. "The Strength of Weak Ties." In: *American Journal of Sociology* (1973).

[82] Mark Granovetter. "The Strength of Weak Ties: A Network Theory Revisited." In: *Sociological Theory* (1983), pp. 201–233.

[83] Reto Grob, Michael Kuhn, Roger Wattenhofer, and Martin Wirz. "Cluestr: Mobile Social Networking for Enhanced Group Communication." In: *Proceedings of the ACM 2009 International Conference on Supporting Group Work (GROUP)*. Sanibel Island, FL, USA: ACM, 2009, pp. 81–90.

[84] Franz Josef Grüneberger, Thomas Heinze, and Pascal Felber. "Adaptive Selective Replication for Complex Event Processing Systems." In: *Proceedings of the First International Workshop on Big Dynamic Distributed Data*. Riva del Garda, Italy: ACM, 2013, pp. 31–36.

[85] Yu Gu, Zhe Zhang, Fan Ye, Hao Yang, Minkyong Kim, Hui Lei, and Zhen Liu. "An Empirical Study of High Availability in Stream Processing Systems." In: *Proceedings of the 10th ACM/IFIP/USENIX International Conference on Middleware*. Urbana Champaign, IL, USA: Springer-Verlag New York, Inc., 2009, p. 23.

[86] Rachid Guerraoui and André Schiper. "Software-Based Replication for Fault Tolerance." In: *Computer* 30.4 (1997), pp. 68–74.

[87] Akhil Gupta and Rakesh Kumar Jha. "A Survey of 5G Network: Architecture and Emerging Technologies." In: *IEEE access* 3 (2015), pp. 1206–1232.

[88] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. "The WEKA Data Mining Software: An Update." In: *ACM SIGKDD Explorations Newsletter* 11.1 (2009).

[89] Russell Hardin. *Trust and Trustworthiness*. Russell Sage Foundation, 2002.

[90] Hannes Hartenstein and LP Laberteaux. "A Tutorial Survey on Vehicular Ad Hoc Networks." In: *IEEE Communications Magazine* 46.6 (2008).

[91] Yeye He, Siddharth Barman, Di Wang, and Jeffrey F Naughton. "On the Complexity of Privacy-Preserving Complex Event Processing." In: *Proceedings of the 30th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. Athens, Greece: ACM, 2011, pp. 165–174.

[92] Thomas Heinze, Valerio Pappalardo, Zbigniew Jerzak, and Christof Fetzer. "Auto-Scaling Techniques for Elastic Data Stream Processing." In: *2014 IEEE 30th International Conference on Data Engineering Workshops (ICDEW)*. IEEE. Chicago, IL, USA, 2014, pp. 296–302.

[93] Thomas Heinze, Mariam Zia, Robert Krahn, Zbigniew Jerzak, and Christof Fetzer. "An Adaptive Replication Scheme for Elastic Data Stream Processing Systems." In: *Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems (DEBS)*. Oslo, Norway: ACM, 2015, pp. 150–161.

[94] Kirak Hong, David Lillethun, Umakishore Ramachandran, Beate Ottenwälder, and Boris Koldehofe. "Mobile Fog: A Programming Model for Large-Scale Applications on the Internet of Things." In: *Proceedings of the Second ACM SIGCOMM Workshop on Mobile Cloud Computing (MCC)*. ACM. Hong Kong, China, 2013, pp. 15–20.

[95] Kirak Hong, David Lillethun, Umakishore Ramachandran, Beate Ottenwälder, and Boris Koldehofe. "Opportunistic Spatio-Temporal Event Processing for Mobile Situation Awareness." In: *Proceedings of the 7th ACM International Conference on Distributed Event-Based Systems (DEBS)*. Arlington, TX, USA: ACM, 2013, pp. 195–206.

[96] Seyed Amir Hoseini-Tabatabaei, Alexander Gluhak, and Rahim Tafazolli. "A Survey on Smartphone-Based Systems for Opportunistic User Context Recognition." In: *ACM Computing Surveys (CSUR)* 45.3 (2013), p. 27.

[97] Darko Hrestak and Stjepan Picek. "Homomorphic Encryption in the Cloud." In: *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE. Opatija, Croatia, 2014, pp. 1400–1404.

[98] Vincent C Hu, D Richard Kuhn, and David F Ferraiolo. "Attribute-Based Access Control." In: *Computer* 48.2 (2015), pp. 85–88.

[99] Vincent C Hu, David Ferraiolo, Rick Kuhn, Arthur R Friedman, Alan J Lang, Margaret M Cogdell, Adam Schnitzer, Kenneth Sandlin, Robert Miller, Karen Scarfone, et al. "Guide to Attribute Based Access Control (ABAC) Definition and Considerations (Draft)." In: *NIST Special Publication* 800.162 (2013).

[100] Jeong-Hyon Hwang, Magdalena Balazinska, Alex Rasin, Ugur Cetintemel, Michael Stonebraker, and Stan Zdonik. "High-Availability Algorithms for Distributed Stream Processing." In: *2005 IEEE 21st International Conference on Data Engineering (ICDE)*. Tokyo, Japan: IEEE, 2005, pp. 779–790.

[101] Jeong-Hyon Hwang, Ying Xing, Ugur Cetintemel, and Stan Zdonik. "A Cooperative, Self-Configuring High-Availability Solution for Stream Processing." In: *2007 IEEE 23rd International Conference on Data Engineering (ICDE)*. Istanbul, Turkey: IEEE, 2007, pp. 176–185.

[102] *Intel® Software Guard Extensions (Intel® SGX)*. URL: `https://software.intel.com/en-us/sgx/details`.

[103] SM Riazul Islam, Daehan Kwak, MD Humaun Kabir, Mahmud Hossain, and Kyung-Sup Kwak. "The Internet of Things for Health Care: A Comprehensive Survey." In: *IEEE Access* 3 (2015), pp. 678–708.

[104] Van Jacobson. "Congestion Avoidance and Control." In: *Symposium Proceedings on Communications Architectures and Protocols (SIGCOMM)*. Stanford, CA, USA: ACM, 1988, pp. 314–329.

[105] Van Jacobson, Diana K Smetters, James D Thornton, Michael F Plass, Nicholas H Briggs, and Rebecca L Braynard. "Networking Named Content." In: *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*. ACM. Rome, Italy, 2009, pp. 1–12.

[106] Mohsen Jamali and Martin Ester. "Trustwalker: A Random Walk Model for Combining Trust-Based and Item-Based Recommendation." In: *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*. ACM. Paris, France, 2009, pp. 397–406.

[107] Kasthuri Jayarajah, Youngki Lee, Archan Misra, and Rajesh Krishna Balan. "Need Accurate User Behaviour?: Pay Attention to Groups!" In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*. ACM. Osaka, Japan, 2015, pp. 855–866.

[108] Sachini Jayasekara, Sameera Kannangara, Tishan Dahanayakage, Isuru Ranawaka, Srinath Perera, and Vishaka Nanayakkara. "Wihidum: Distributed Complex Event Processing." In: *Journal of Parallel and Distributed Computing* 79 (2015), pp. 42–51.

[109] Aravind J Joshi. "Natural Language Processing." In: *Science* (1991), pp. 1242–1249.

[110] Piotr Kamisiński, Vera Goebel, and Thomas Plagemann. "A Reconfigurable Distributed CEP Middleware for Diverse Mobility Scenarios." In: *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. IEEE. San Diego, CA, USA, 2013, pp. 615–620.

[111] Sepandar D. Kamvar, Mario T. Schlosser, and Hector Garcia-Molina. "The Eigentrust Algorithm for Reputation Management in P2P Networks." In: *Proceedings of the 12th International Conference on World Wide Web (WWW)*. Budapest, Hungary: ACM, 2003, pp. 640–651.

[112] Sebastian Kaune. "Performance and Availability in Peer-to-Peer Content Distribution Systems: A Case for a Multilateral Incentive Approach." PhD Thesis. TU Darmstadt, 2011.

[113] Kari Kelton, Kenneth R Fleischmann, and William A Wallace. "Trust in Digital Information." In: *Journal of the Association for Information Science and Technology* 59.3 (2008), pp. 363–374.

[114] Ashraf Khalil and Kay Connelly. "Context-aware Telephony: Privacy Preferences and Sharing Patterns." In: *Proceedings of the 2006 20th Anniversary Conference on Computer Supported Cooperative Work (CSCW)*. Banff, AB, Canada: ACM, 2006, pp. 469–478.

[115] Valerie King and Jared Saia. "Breaking the $O(n^2)$ Bit Barrier: Scalable Byzantine Agreement with an Adaptive Adversary." In: *Journal of the ACM (JACM)* 58.4 (2011), p. 18.

[116] Boris Koldehofe, Ruben Mayer, Umakishore Ramachandran, Kurt Rothermel, and Marco Völz. "Rollback-Recovery without Checkpoints in Distributed Event Processing Systems." In: *Proceedings of the 7th ACM International Conference on Distributed Event-Based Systems (DEBS)*. Arlington, TX, USA: ACM, 2013, pp. 27–38.

[117] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas. "Machine Learning: A Review of Classification and Combining Techniques." In: *Artificial Intelligence Review* 26.3 (2006), pp. 159–190.

[118] Aleksandra Kovačević. "Peer-to-Peer Location-Based Search: Engineering a Novel Peer-to-Peer Overlay Network." PhD Thesis. TU Darmstadt, 2009.

[119] Matthias Kropff. "Sensor-basiertes Monitoring zur kontextsensitiven Unterstützung von Wissensarbeit." PhD Thesis. TU Darmstadt, 2010.

[120] Mariantonietta La Polla, Fabio Martinelli, and Daniele Sgandurra. "A Survey on Security for Mobile Devices." In: *IEEE Communications Surveys & Tutorials* 15.1 (2013), pp. 446–471.

[121] Geetika T. Lakshmanan, Ying Li, and Rob Strom. "Placement Strategies for Internet-Scale Data Stream Systems." In: *IEEE Internet Computing* 12.6 (2008), pp. 50–60.

[122] Peng Li and Wang Bingwen. "Design of Complex Event Processing System for Wireless Sensor Networks." In: *2010 Second International Conference on Networks Security Wireless Communications and Trusted Computing (NSWCTC)*. Vol. 1. IEEE. Wuhan, China, 2010, pp. 354–357.

[123] Yong Li, Ting Wu, Pan Hui, Depeng Jin, and Sheng Chen. "Social-aware D2D Communications: Qualitative Insights and Quantitative Analysis." In: *IEEE Communications Magazine* 52.6 (2014), pp. 150–158.

[124] Nicolas Christopher Liebau. "Trusted Accounting in Peer-to-Peer Environments: A Novel Token-based Accounting Scheme for Autonomous Distributed Systems." PhD Thesis. TU Darmstadt, 2008.

[125] Nan Lin, Walter M Ensel, and John C Vaughn. "Social Resources and Strength of Ties: Structural Factors in Occupational Status Attainment." In: *American Sociological Review* (1981), pp. 393–405.

[126] Wolfgang Lindner and Jorg Meier. "Securing the Borealis Data Stream Engine." In: *10th International Conference on Database Engineering and Applications Symposium (IDEAS)*. IEEE. Delhi, India, 2006, pp. 137–147.

[127] Haifeng Liu, Ee-Peng Lim, Hady W. Lauw, Minh-Tam Le, Aixin Sun, Jaideep Srivastava, and Young Ae Kim. "Predicting Trusts Among Users of Online Communities: An Epinions Case Study." In: *Proceedings of the 9th ACM Conference on Electronic Commerce*. Chicago, IL, USA: ACM, 2008, pp. 310–319.

[128] Seng Wai Loke. "On Representing Situations for Context-Aware Pervasive Computing: Six Ways to Tell if You Are in a Meeting." In: *Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, 2006 (PerCom Workshops)*. Pisa, Italy: IEEE, 2006, pp. 35–39.

[129] Hong Lu, Denise Frauendorfer, Mashfiqui Rabbi, Marianne Schmid Mast, Gokul T Chittaranjan, Andrew T Campbell, Daniel Gatica-Perez, and Tanzeem Choudhury. "Stresssense: Detecting Stress in Unconstrained Acoustic Environments Using Smartphones." In: *Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp)*. Pittsburgh, PA, USA: ACM, 2012, pp. 351–360.

[130] Alexandra Marin and Keith N Hampton. "Simplifying the Personal Network Name Generator Alternatives to Traditional Multiple and Single Name Generators." In: *Field Methods* 19.2 (2007).

[131] Gloria Mark, Daniela Gudith, and Ulrich Klocke. "The Cost of Interrupted Work: More Speed and Stress." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. Florence, Italy: ACM, 2008, pp. 107–110.

[132] Peter V Marsden and Karen E Campbell. "Measuring Tie Strength." In: *Social Forces* 63.2 (1984).

[133] Roger C Mayer, James H Davis, and F David Schoorman. "An Integrative Model of Organizational Trust." In: *Academy of Management Review* 20.3 (1995), pp. 709–734.

[134] Ruben Mayer, Boris Koldehofe, and Kurt Rothermel. "Predictable Low-Latency Event Detection with Parallel Complex Event Processing." In: *IEEE Internet of Things Journal* 2.4 (2015), pp. 274–286.

[135] Afef Mdhaffar, Tarak Chaari, Kaouthar Larbi, Mohamed Jmaiel, and Bernd Freisleben. "IoT-Based Health Monitoring via LoRaWAN." In: *17th IEEE International Conference on Smart Technologies (EUROCON)*. IEEE. Ohrid, Macedonia, 2017, pp. 519–524.

[136] Afef Mdhaffar, Riadh Ben Halima, Mohamed Jmaiel, and Bernd Freisleben. "CEP4Cloud: Complex Event Processing for Self-Healing Clouds." In: *2014 IEEE 23rd International WETICE Conference (WETICE)*. IEEE. Parma, Italy, 2014, pp. 62–67.

[137] Abhinav Mehrotra, Mirco Musolesi, Robert Hendley, and Veljko Pejovic. "Designing Content-driven Intelligent Notification Mechanisms for Mobile Applications." In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*. Osaka, Japan: ACM, 2015, pp. 813–824.

[138] Sarah Mennicken, Jo Vermeulen, and Elaine M. Huang. "From Today's Augmented Houses to Tomorrow's Smart Homes: New Directions for Home Automation Research." In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*. Seattle, Washington: ACM, 2014, pp. 105–115.

[139] Matteo Migliavacca, Ioannis Papagiannis, David M Eyers, Brian Shand, Jean Bacon, and Peter Pietzuch. "DEFCon: High-Performance Event Processing with Information Security." In: *Proceedings of the 2010 USENIX Annual Technical Conference (ATC)*. USENIX Association. Boston, MA, USA, 2010, pp. 1–15.

[140] Leonardo Militano, Giuseppe Araniti, Massimo Condoluci, Ivan Farris, and Antonio Iera. "Device-to-Device Communications for 5G Internet of Things." In: *EAI Endorsed Transactions on Internet of Things* 1.1 (2015), pp. 1–15.

[141] Leonardo Militano, Antonino Orsino, Giuseppe Araniti, and Antonio Iera. "NB-IoT for D2D-Enhanced Content Uploading with Social Trustworthiness in 5G Systems." In: *Future Internet* 9.3 (2017), 31:1–14.

[142] Jun-Ki Min, Jason Wiese, Jason Hong, and John Zimmerman. "Mining Smartphone Data to Classify Life-Facets of Social Relationships." In: *Proceedings of the 2013 Conference on Computer Supported Cooperative Work (CSCW)*. San Antonio, TX, USA: ACM, 2013, pp. 285–294.

[143] Yves-Alexandre de Montjoye, Jordi Quoidbach, Florent Robic, and Alex Sandy Pentland. "Predicting Personality Using Novel Mobile Phone-Based Metrics." In: *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction (SBP)*. Washington, DC, USA: Springer-Verlag, 2013, pp. 48–55.

[144] Lik Mui, Mojdeh Mohtashemi, and Ari Halberstadt. "A Computational Model of Trust and Reputation." In: *Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS)*. IEEE. Big Island, HI, USA, 2002, pp. 2431–2439.

[145] Christopher Mutschler and Michael Philippsen. "Runtime Migration of Stateful Event Detectors with Low-Latency Ordering Constraints." In: *2013 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. IEEE. San Diego, CA, USA, 2013, pp. 609–614.

[146] Mohamed Nabeel, Stefan Appel, Elisa Bertino, and Alejandro Buchmann. "Privacy Preserving Context Aware Publish Subscribe Systems." In: *International Conference on Network and System Security*. Springer. Madrid, Spain, 2013, pp. 465–478.

[147] Mohamed Nabeel, Ning Shang, and Elisa Bertino. "Efficient Privacy Preserving Content Based Publish Subscribe Systems." In: *Proceedings of the 17th ACM Symposium on Access Control Models and Technologies (SACMAT)*. ACM. Newark, NJ, USA, 2012, pp. 133–144.

[148] Svenja Neitzel, Frank Englert, Rahul Dwarakanath, Katharina Schneider, Kathrin Reinke, Gisela Gerlach, Christoph Rensing, Doreen Boehnstedt, and Ruth Stock-Homburg. "Towards Using Situational Information to Detect an Individual's Perceived Stress Level." In: *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. Kona, HI, USA: IEEE, 2017, pp. 333–338.

[149] Surya Nepal, Wanita Sherchan, and Cecile Paris. "STrust: A Trust Model for Social Networks." In: *2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE. Changsha, China, 2011, pp. 841–846.

[150] The An Binh Nguyen, Pratyush Agnihotri, Christian Meurisch, Manisha Luthra, Rahul Dwarakanath, Jeremias Blendin, Doreen Boehnstedt, Michael Zink, and Ralf Steinmetz. "Efficient Crowd Sensing Task Distribution Through Context-aware NDN-based Geocast." In: *42nd IEEE International Conference on Local Computer Networks (LCN)*. Singapore, Singapore: IEEE, 2017, pp. 52–60.

[151] Christena E Nippert-Eng. *Home and Work: Negotiating Boundaries Through Everyday Life*. University of Chicago Press, 2008.

[152] Anderson Santana de Oliveira, Florian Kerschbaum, Hoon Wei Lim, and Su-Yang Yu. "Privacy-Preserving Techniques and System for Streaming Databases." In: *2012 International Conference on Privacy, Security, Risk and Trust (PASSAT) and 2012 International Conference on Social Computing (SocialCom)*. IEEE. Amsterdam, Netherlands, 2012, pp. 728–733.

[153] Judith S Olson, Jonathan Grudin, and Eric Horvitz. "A Study of Preferences for Sharing and Privacy." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI): Extended Abstracts*. Portland, OR, USA: ACM, 2005, pp. 1985–1988.

[154] Aleksandr Ometov, Ekaterina Olshannikova, Pavel Masek, Thomas Olsson, Jiri Hosek, Sergey Andreev, and Yevgeni Koucheryavy. "Dynamic Trust Associations Over Socially-Aware D2D Technology: A Practical Implementation Perspective." In: *IEEE Access* 4 (2016), pp. 7692–7702.

[155] Emanuel Onica, Pascal Felber, Hugues Mercier, and Etienne Rivière. "Confidentiality-Preserving Publish/Subscribe: A Survey." In: *ACM Computing Surveys (CSUR)* 49.2 (2016), p. 27.

[156] J-P Onnela, Jari Saramäki, Jorkki Hyvönen, György Szabó, David Lazer, Kimmo Kaski, János Kertész, and A-L Barabási. "Structure and Tie Strengths in Mobile Communication Networks." In: *Proceedings of the National Academy of Sciences* 104.18 (2007).

[157] Beate Ottenwälder, Boris Koldehofe, Kurt Rothermel, Kirak Hong, David Lillethun, and Umakishore Ramachandran. "MCEP: A Mobility-Aware Complex Event Processing System." In: *ACM Transactions on Internet Technology (TOIT)* 14.1 (2014), p. 6.

[158] Beate Ottenwälder, Boris Koldehofe, Kurt Rothermel, Kirak Hong, and Umakishore Ramachandran. "RECEP: Selection-Based Reuse for Distributed Complex Event Processing." In: *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems (DEBS)*. ACM. Mumbai, India, 2014, pp. 59–70.

[159] Beate Ottenwälder, Boris Koldehofe, Kurt Rothermel, and Umakishore Ramachandran. "MigCEP: Operator Migration for Mobility Driven Distributed Complex Event Processing." In: *Proceedings of the 7th ACM International Conference on Distributed Event-Based Systems (DEBS)*. Arlington, TX, USA: ACM, 2013, pp. 183–194.

[160] Fatih Kursat Ozenc and Shelly D Farnham. "Life Modes in Social Media." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. ACM. Vancouver, BC, Canada, 2011, pp. 561–570.

[161] Jeongyeup Paek, Joongheon Kim, and Ramesh Govindan. "Energy-efficient Rate-adaptive GPS-based Positioning for Smartphones." In: *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys)*. San Francisco, CA, USA: ACM, 2010, pp. 299–314.

[162] Pascal Paillier. "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes." In: *International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*. Vol. 99. Springer. Prague, Czech Republic, 1999, pp. 223–238.

[163] Helge Parzyjegla, Daniel Graff, Arnd Schröter, Jan Richling, and Gero Mühl. "Design and Implementation of the Rebeca Publish/Subscribe Middleware." In: *From Active Data Management to Event-Based Systems and More* 6462 (2010), pp. 124–140.

[164] Martin Pielot, Karen Church, and Rodrigo de Oliveira. "An In-situ Study of Mobile Phone Notifications." In: *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services (MobileHCI)*. Toronto, ON, Canada: ACM, 2014, pp. 233–242.

[165] Peter R Pietzuch and Jean M Bacon. "Hermes: A Distributed Event-Based Middleware Architecture." In: *Proceedings of the 22nd International Conference on Distributed Computing Systems Workshops*. IEEE. Vienna, Austria, 2002, pp. 611–618.

[166] Peter R Pietzuch, Brian Shand, and Jean Bacon. "Composite Event Detection as a Generic Middleware Extension." In: *IEEE Network* 18.1 (2004), pp. 44–55.

[167] Peter Pietzuch, Jonathan Ledlie, Jeffrey Shneidman, Mema Roussopoulos, Matt Welsh, and Margo Seltzer. "Network-Aware Operator Placement for Stream-Processing Systems." In: *Proceedings of the 22nd International Conference on Data Engineering (ICDE)*. IEEE. 2006, pp. 49–49.

[168] Daniele Quercia, Stephen Hailes, and Licia Capra. "B-trust: Bayesian Trust Framework for Pervasive Computing." In: *Lecture Notes in Computer Science* 3986 (2006), pp. 298–312.

[169] Maxim Raya, Reza Shokri, and Jean-Pierre Hubaux. "On the Tradeoff Between Trust and Privacy in Wireless Ad Hoc Networks." In: *Proceedings of the Third ACM Conference on Wireless Network Security (WiSec)*. Hoboken, NJ, USA: ACM, 2010, pp. 75–80.

[170] Delphine Reinhardt, Franziska Engelmann, and Matthias Hollick. "Can I Help You Setting Your Privacy? A Survey-based Exploration of Users' Attitudes Towards Privacy Suggestions." In: *Proceedings of the 13th International Conference on Advances in Mobile Computing and Multimedia (MoMM)*. Brussels, Belgium: ACM, 2015, pp. 347–356.

[171] Delphine Reinhardt, Franziska Engelmann, Andrey Moerov, and Matthias Hollick. "Show Me Your Phone, I Will Tell You Who Your Friends are: Analyzing Smartphone Data to Identify Social Relationships." In: *Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia (MUM)*. Linz, Austria: ACM, 2015, pp. 75–83.

[172] John K Rempel, John G Holmes, and Mark P Zanna. "Trust in Close Relationships." In: *Journal of Personality and Social Psychology* 49.1 (1985), p. 95.

[173] Robert Remus, Uwe Quasthoff, and Gerhard Heyer. "SentiWS-A Publicly Available German-language Resource for Sentiment Analysis." In: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC)*. Valletta, Malta, 2010.

[174] Kui Ren, Tieyan Li, Zhiguo Wan, Feng Bao, Robert H. Deng, and Kwangjo Kim. "Highly Reliable Trust Establishment Scheme in Ad Hoc Networks." In: *Computer Networks* 45.6 (2004), pp. 687–699.

[175] Sebastian Ries. "Extending Bayesian Trust Models Regarding Context-Dependence and User Friendly Representation." In: *Proceedings of the 2009 ACM Symposium on Applied Computing (SAC)*. ACM. Honolulu, HI, USA, 2009, pp. 1294–1301.

[176] Stamatia Rizou, Frank Diirr, and Kurt Rothermel. "Fulfilling End-to-End Latency Constraints in Large-Scale Streaming Environments." In: *2011 IEEE 30th International Performance Computing and Communications Conference (IPCCC)*. IEEE. Orlando, FL, USA, 2011, pp. 1–8.

[177] Sam GB Roberts and Robin IM Dunbar. "Communication in Social Networks: Effects of Kinship, Network Size, and Emotional Closeness." In: *Personal Relationships* 18.3 (2011).

[178] Sarah León Rojas. "Analyse sozialer Beziehungen anhand nonverbaler Signale im textbasierten Chat." Master Thesis. Cologne University of Applied Sciences, 2011.

[179] Sarah León Rojas, Uwe Kirschenmann, and Martin Wolpers. "We Have no Feelings, We Have Emoticons ;-)." In: *IEEE 12th International Conference on Advanced Learning Technologies (ICALT)*. Rome, Italy: IEEE, 2012, pp. 642–646.

[180] Kay Römer. "Time Synchronization in Ad Hoc Networks." In: *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc)*. ACM, 2001, pp. 173–182.

[181] Atanu Roy, Ayush Singhal, and Jaideep Srivastava. "Formation and Reciprocation of Dyadic Trust." In: *ACM Transactions on Internet Technology (TOIT)* 17.2 (2017), 15:1–15:24.

[182] Alireza Sahami Shirazi, Niels Henze, Tilman Dingler, Martin Pielot, Dominik Weber, and Albrecht Schmidt. "Large-scale Assessment of Mobile Notifications." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*. Toronto, Ontario, Canada: ACM, 2014, pp. 3055–3064.

[183] Omran Saleh and Kai-Uwe Sattler. "Distributed Complex Event Processing in Sensor Networks." In: *2013 IEEE 14th International Conference on Mobile Data Management (MDM)*. Vol. 2. IEEE. Milan, Italy, 2013, pp. 23–26.

[184] Akane Sano and Rosalind W Picard. "Stress Recognition Using Wearable Sensors and Mobile Phones." In: *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction (ACII)*. Geneva, Switzerland: IEEE, 2013, pp. 671–676.

[185] Mahadev Satyanarayanan, Paramvir Bahl, Ramón Caceres, and Nigel Davies. "The Case for VM-Based Cloudlets in Mobile Computing." In: *IEEE Pervasive Computing* 8.4 (2009).

[186] Wilmar B Schaufeli, Michael P Leiter, and Christina Maslach. "Burnout: 35 Years of Research and Practice." In: *Career Development International* 14.3 (2009).

[187] Björn Schilling, Boris Koldehofe, Kurt Rothermel, and Umakishore Ramachandran. "Access Policy Consolidation for Event Processing Systems." In: *2013 Conference on Networked Systems (NetSys)*. Stuttgart, Germany: IEEE, 2013, pp. 92–101.

[188] Fred B. Schneider. "Implementing Fault-Tolerant Services using the State Machine Approach: A Tutorial." In: *ACM Computing Surveys (CSUR)* 22.4 (1990), pp. 299–319.

[189] Zoe Sebepou and Kostas Magoutis. "CEC: Continuous Eventual Checkpointing for Data Stream Processing Operators." In: *2011 IEEE/IFIP 41st International Conference on Dependable Systems & Networks (DSN)*. Hong Kong, China: IEEE, 2011, pp. 145–156.

[190] Wanita Sherchan, Surya Nepal, and Cecile Paris. "A Survey of Trust in Social Networks." In: *ACM Computing Surveys* 45.4 (2013), 47:1–47:33.

[191] Justine Sherry, Peter Xiang Gao, Soumya Basu, Aurojit Panda, Arvind Krishnamurthy, Christian Maciocco, Maziar Manesh, João Martins, Sylvia Ratnasamy, Luigi Rizzo, and Scott Shenker. "Rollback-Recovery for Middleboxes." In: *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication (SIGCOMM)*. London, United Kingdom: ACM, 2015, pp. 227–240.

[192] Abdullatif Shikfa, Melek Önen, and Refik Molva. "Privacy-Preserving Content-Based Publish/Subscribe Networks." In: *24th International Conference on Information Security and Privacy Protection (SEC)*. Springer. Pafos, Cyprus, 2009, pp. 270–282.

[193] Sabrina Sicari, Alessandra Rizzardi, Luigi Alfredo Grieco, and Alberto Coen-Porisini. "Security, Privacy and Trust in Internet of Things: The Road Ahead." In: *Computer Networks* 76 (2015), pp. 146–164.

[194] Jatinder Singh, David M Eyers, and Jean Bacon. "Disclosure Control in Multi-Domain Publish/Subscribe Systems." In: *Proceedings of the 5th ACM International Conference on Distributed Event-Based Systems (DEBS)*. ACM. New York, NY, USA, 2011, pp. 159–170.

[195] Jatinder Singh, Thomas Pasquier, Jean Bacon, Julia Powles, Raluca Diaconu, and David Eyers. "Big Ideas Paper: Policy-Driven Middleware for a Legally-Compliant Internet of Things." In: *Proceedings of the 17th International Middleware Conference*. Trento, Italy: ACM, 2016, p. 13.

[196] Jatinder Singh, Luis Vargas, and Jean Bacon. "A Model for Controlling Data Flow in Distributed Healthcare Environments." In: *Second International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth)*. IEEE. Tampere, Finland, 2008, pp. 188–191.

[197] Jatinder Singh, Luis Vargas, Jean Bacon, and Ken Moody. "Policy-Based Information Sharing in Publish/Subscribe Middleware." In: *IEEE Workshop on Policies for Distributed Systems and Networks (POLICY)*. IEEE. Palisades, NY, USA, 2008, pp. 137–144.

[198] Daniel Specht, Benjamin Schiller, and Marius Rettberg-Päplow. "Activity Recognition Based on User Interaction Patterns on Smartphones." Lab Project. TU Darmstadt, 2017.

[199] Tasos Spiliotopoulos, Diogo Pereira, and Ian Oakley. "Predicting Tie Strength with the Facebook API." In: *Proceedings of the 18th Panhellenic Conference on Informatics*. Athens, Greece: ACM, 2014, pp. 1–5.

[200] Mudhakar Srivatsa, Li Xiong, and Ling Liu. "TrustGuard: Countering Vulnerabilities in Reputation Management for Decentralized Overlay Networks." In: *Proceedings of the 14th international conference on World Wide Web (WWW)*. Chiba, Japan: ACM, 2005, pp. 422–431.

[201] Fabrice Starks and Thomas Peter Plagemann. "Operator Placement for Efficient Distributed Complex Event Processing in MANETs." In: *2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. Abu Dhabi, UAE: IEEE, 2015, pp. 83–90.

[202] Ralf Steinmetz and Klaus Wehrle. *Peer-to-Peer Systems and Applications*. Vol. 3485. Lecture Notes in Computer Science. Springer, 2005.

[203] Marco Stolpe. "The Internet of Things: Opportunities and Challenges for Distributed Data Analysis." In: *ACM SIGKDD Explorations Newsletter* 18.1 (2016), pp. 15–34.

[204] Li Su and Yongluan Zhou. "Tolerating Correlated Failures in Massively Parallel Stream Processing Engines." In: *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. Helsinki, Finland: IEEE, 2016, pp. 517–528.

[205] Sriskandarajah Suhothayan, Kasun Gajasinghe, Isuru Loku Narangoda, Subash Chaturanga, Srinath Perera, and Vishaka Nanayakkara. "Siddhi: A Second Look at Complex Event Processing Architectures." In: *Proceedings of the 2011 ACM Workshop on Gateway Computing Environments (GCE)*. Seattle, WA, USA: ACM, 2011, pp. 43–50.

[206] Yan Lindsay Sun, Wei Yu, Zhu Han, and KJ Ray Liu. "Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks." In: *IEEE Journal on Selected Areas in Communications* 24.2 (2006), pp. 305–317.

[207] Piotr Sztompka. *Trust: A Sociological Theory*. Cambridge University Press, 1999.

[208] Ugur Eray Tahta, Sevil Sen, and Ahmet Burak Can. "GenTrust: A Genetic Trust Management Model for Peer-to-Peer Systems." In: *Applied Soft Computing* 34 (2015), pp. 693–704.

[209] Henri Tajfel. *Social Identity and Intergroup Relations*. Cambridge University Press, 2010.

[210] Muhammad Adnan Tariq, Boris Koldehofe, and Kurt Rothermel. "Securing Broker-less Publish/Subscribe Systems using Identity-Based Encryption." In: *IEEE Transactions on Parallel and Distributed Systems* 25.2 (2014), pp. 518–528.

[211] Mohsen Nader Tehrani, Murat Uysal, and Halim Yanikomeroglu. "Device-to-Device Communication in 5G Cellular Networks: Challenges, Solutions, and Future Directions." In: *IEEE Communications Magazine* 52.5 (2014), pp. 86–92.

[212] Sebastian Trapp, Matthias Wählisch, and Jochen H. Schiller. "Short Paper: Can Your Phone Trust Your Friend Selection?" In: *Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices (SPSM)*. Chicago, IL, USA: ACM, 2011, pp. 69–74.

[213] Sacha Trifunovic, Franck Legendre, and Carlos Anastasiades. "Social Trust in Opportunistic Networks." In: *INFOCOM IEEE Conference on Computer Communications Workshops*. IEEE. San Diego, CA, USA, 2010, pp. 1–6.

[214] Luis M Vaquero and Luis Rodero-Merino. "Finding Your Way in the Fog: Towards a Comprehensive Definition of Fog Computing." In: *ACM SIGCOMM Computer Communication Review* 44.5 (2014), pp. 27–32.

[215] Tim Verbelen, Pieter Simoens, Filip De Turck, and Bart Dhoedt. "Cloudlets: Bringing the Cloud to the Mobile User." In: *Proceedings of the Third ACM Workshop on Mobile Cloud Computing and Services (MCS)*. Low Wood Bay, UK: ACM, 2012, pp. 29–36.

[216] BS Vidyalakshmi, Raymond K Wong, and Chi-Hung Chi. "Decentralized Trust Driven Access Control for Mobile Content Sharing." In: *2013 IEEE International Congress on Big Data (BigData Congress)*. Santa Clara, CA, USA: IEEE, 2013, pp. 239–246.

[217] Marco Völz, Boris Koldehofe, and Kurt Rothermel. "Supporting Strong Reliability for Distributed Complex Event Processing Systems." In: *2011 IEEE 13th International Conference on High Performance Computing and Communications (HPCC)*. Banff, AB, Canada: IEEE, 2011, pp. 477–486.

[218] Joseph B Walther and Kyle P D'Addario. "The Impacts of Emoticons on Message Interpretation in Computer-Mediated Communication." In: *Social Science Computer Review* 19.3 (2001), pp. 324–347.

[219] Huayong Wang and Li-Shiuan Peh. "MobiStreams: A Reliable Distributed Stream Processing System for Mobile Devices." In: *2014 IEEE 28th International Parallel and Distributed Processing Symposium (IPDPS)*. Phoenix, AZ, USA: IEEE, 2014, pp. 51–60.

[220] Yang Wang, Gregory Norcie, Saranga Komanduri, Alessandro Acquisti, Pedro Giovanni Leon, and Lorrie Faith Cranor. ""I Regretted the Minute I Pressed Share": A Qualitative Study of Regrets on Facebook." In: *Proceedings of the Seventh Symposium on Usable Privacy and Security (SOUPS)*. Pittsburgh, PA, USA: ACM, 2011, 10:1–10:16.

[221] Yi Wang and David Redmiles. "The Diffusion of Trust and Cooperation in Teams with Individuals' Variations on Baseline Trust." In: *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing (CSCW)*. ACM. San Francisco, CA, USA, 2016, pp. 303–318.

[222] Barry Wellman and Scot Wortley. "Different Strokes from Different Folks: Community Ties and Social Support." In: *American Journal of Sociology* 96.3 (1990), pp. 558–588.

[223] WhatsApp. *Cumulative Daily Mobile Message Volume of WhatsApp Messenger as of April 2014 (in Billions)*. 2014. URL: `http://www.statista.com/statistics/258743/daily-mobile-message-volume-of-whatsapp-messenger/`.

[224] WhatsApp. *Number of Monthly Active WhatsApp Users Worldwide from April 2013 to January 2017 (in Millions)*. 2017. URL: `http://www.statista.com/statistics/260819/number-of-monthly-active-whatsapp-users/`.

[225] Jason Wiese, Patrick Gage Kelley, Lorrie Faith Cranor, Laura Dabbish, Jason I. Hong, and John Zimmerman. "Are You Close with Me? Are You Nearby?: Investigating Social Groups, Closeness, and Willingness to Share." In: *Proceedings of the 13th International Conference on Ubiquitous Computing (UbiComp)*. Beijing, China: ACM, 2011, pp. 197–206.

[226] Jason Wiese, Jun-Ki Min, Jason I Hong, and John Zimmerman. *Assessing Call and SMS Logs as an Indication of Tie Strength*. Technical Report. Carnegie Mellon University, 2014.

[227] Jason Wiese, Jun-Ki Min, Jason I Hong, and John Zimmerman. "You Never Call, You Never Write: Call and SMS Logs Do Not Always Indicate Tie Strength." In: *Proceedings of the 2015 Conference on Computer Supported Cooperative Work (CSCW)*. Vancouver, BC, Canada: ACM, 2015, pp. 765–774.

[228] Dan Wu, Liang Zhou, and Yueming Cai. "Social-Aware Rate Based Content Sharing Mode Selection for D2D Content Sharing Scenarios." In: *IEEE Transactions on Multimedia* PP.99 (2017), pp. 1–12.

[229] Li Xiong and Ling Liu. "PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities." In: *IEEE Transactions on Knowledge and Data Engineering* 16.7 (2004), pp. 843–857.

[230] Hao Yang, Haiyun Luo, Fan Ye, Songwu Lu, and Lixia Zhang. "Security in Mobile Ad Hoc Networks: Challenges and Solutions." In: *IEEE Wireless Communications* 11.1 (2004), pp. 38–47.

[231] Jin Yang, Tianli Mo, Lipyeow Lim, Kai-Uwe Sattler, and Archan Misra. "Energy-Efficient Collaborative Query Processing Framework for Mobile Sensing Services." In: *2013 IEEE 14th International Conference on Mobile Data Management (MDM)*. Milan, Italy: IEEE, 2013, pp. 147–156.

[232] Jinhui Yao, Shiping Chen, Surya Nepal, David Levy, and John Zic. "TrustStore: Making Amazon S3 Trustworthy with Services Composition." In: *Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGRID)*. Melbourne, Australia: IEEE Computer Society, 2010, pp. 600–605.

[233] Bin Yu and Munindar P. Singh. "A Social Mechanism of Reputation Management in Electronic Communities." In: *International Workshop on Cooperative Information Agents*. Boston, MA, USA: Springer, 2000, pp. 154–165.

[234] Mo Yu, Wenjun Si, Guojie Song, Zhenhui Li, and John Yen. "Who Were You Talking to – Mining Interpersonal Relationships from Cellphone Network Data." In: *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. Beijing, China: IEEE, 2014, pp. 485–490.

[235] Shucheng Yu, Cong Wang, Kui Ren, and Wenjing Lou. "Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing." In: *2010 Proceedings IEEE INFOCOM*. IEEE. San Diego, CA, USA, 2010, pp. 1–9.

[236] Matei Zaharia, Tathagata Das, Haoyuan Li, Timothy Hunter, Scott Shenker, and Ion Stoica. "Discretized Streams: Fault-Tolerant Streaming Computation at Scale." In: *Proceedings of the 24th ACM Symposium on Operating Systems Principles*. Farminton, PA, USA: ACM, 2013, pp. 423–438.

[237] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. "Internet of Things for Smart Cities." In: *IEEE Internet of Things Journal* 1.1 (2014), pp. 22–32.

[238] S. Zhao, N. Du, A. Nauerz, X. Zhang, Q. Yuan, and R. Fu. "Improved Recommendation Based on Collaborative Tagging Behaviors." In: *Proceedings of the 13th International Conference on Intelligent User Interfaces (IUI)*. Gran Canaria, Spain: ACM, 2008, pp. 413–416.

All web pages cited in this work were last accessed in October 2017. However, due to the dynamic nature of the World Wide Web, their long-term availability cannot be guaranteed.

A

APPENDIX

A.1 COMPARISON OF SUPERVISED MACHINE LEARNING ALGORITHMS

As part of the estimation of the social circle and strength of user relationships, we employed supervised machine learning algorithms to determine the correlation between the extracted features and the ground truth provided by the users in our field studies, *Phase I* and *Phase II*. We analyzed the performance of four popular machine learning algorithms for binary classification—Support Vector Machines (SVM), rule-based decision tree models C4.5 and Random Forest (RF), and the probabilistic model, Näive Bayes (NB). For the further analysis of the obtained datasets, we tested these four algorithms using the 10-fold cross-validation approach, as described in Section 4.3.1.2.

For the subsequent analysis, we used the performance metrics: accuracy and Cohen's Kappa κ. Accuracy provides a measure of the number of correctly estimated instances among all instances in the dataset. Cohen's Kappa $\kappa \in [0, 1]$ provides a measure of the performance improvement that the algorithm provides with respect to the given dataset, in comparison to a random agreement. We measure the accuracy and κ values for each of the above algorithms with respect to the four main social circles—friend, family, work, and hobby—to determine the algorithm that performs best with respect to our dataset.

Table 18: Comparison of the Four Main Machine Learning Algorithms w.r.t. our Dataset in *Phase I*

	Accuracy (%)				**Kappa**			
	J48	*LibSVM*	*NB*	*RF*	*J48*	*LibSVM*	*NB*	*RF*
Friend	55.47	59.11	58.30	62.35	0.10	0.18	0.15	0.25
Family	65.75	63.01	64.38	65.75	0.31	0.26	0.28	0.32
Work	64.86	62.16	64.86	78.38	0.26	0.23	0.31	0.55
Hobby	73.61	56.25	65.28	67.36	0.47	0.12	0.31	0.35
Average	**64.92**	**60.13**	**63.21**	**68.46**	**0.29**	**0.20**	**0.26**	**0.36**
Std. Dev.	**6.43**	**2.67**	**2.85**	**6.01**	**0.13**	**0.05**	**0.07**	**0.11**

Table 18 presents the results of the comparison. We evaluated each of the abovementioned machine learning algorithms using their equivalent implementations using the machine learning tool WEKA. Accordingly, we used the *LibSVM* implementation for SVM; the *J48* implementation for C4.5; and the namesake implementations for RF and NB.

From the results in Table 18, we can clearly see that the performance of the *RF* algorithm is superior to that of the other algorithms, both in terms of accuracy as well as κ. It achieves an average of 5.71% improvement over the other algorithms in accuracy, and a higher κ by 0.11. While the standard deviation of the accuracy across the four social circles is higher than that of *LibSVM* and *NB*, *RF* performs better than these algorithms with respect to all social circles. Consequently, we decided to proceed with the estimation of the user relationships using the *RF* algorithm provided by WEKA.

A.2 ANALYSIS OF TRUSTCEP AGAINST ON-OFF ATTACKS (CONTINUED)

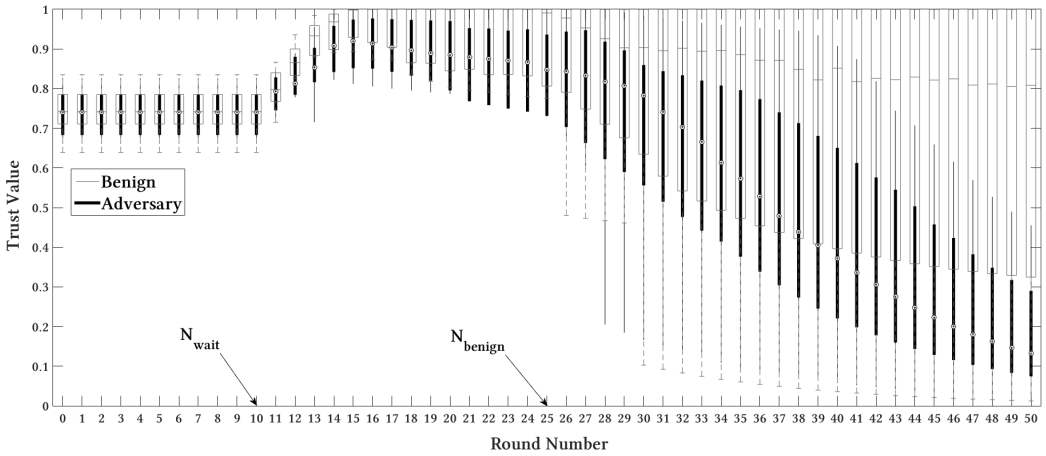
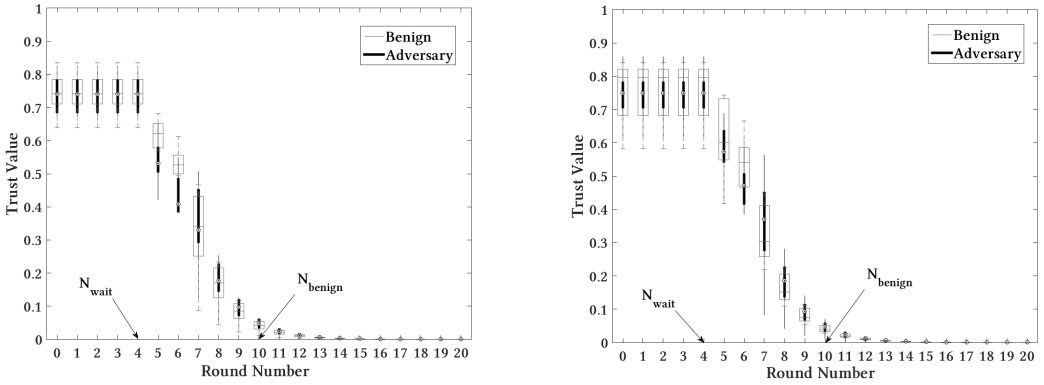


Figure 37: Trust Value Distribution upon On-Off Attack with Increase in Number of Rounds Minority Adversary Population: 20 Users, 12 Benign, 6 Adversary

In this section, we show some results from our analytical evaluations of TrustCEP with respect to on-off attacks. We particularly focus on the analysis of the trust level distribution for different trust modification coefficients. Just to recall, we model on-off attacks such that adversaries behave benignly for a fixed period of time called n_{benign}, before turning malicious. Figure 37 illustrates the distribution of the trust levels, from the benign users' point of view, for a sample case with on-off attacks, where the adversaries are in the minority. We observe a clear increase in the trust values obtained by the adversaries after the initial waiting period, $n_{wait} = 10$, elapses. However, after the adversaries turn malicious after $n_{benign} = 25$, we observe a rapid decline in their obtained trust values. We also observe that the initial rise in the trust values of the adversaries has led to a decrease of trust value for some of the benign users. However, more than 50% of the benign users obtain a trust value of 0.8 or higher.

In the above figure, we used the default values for the trust modification coefficients, $\tau_{inc} = 0.05$ and $\tau_{dec} = 0.75$. In the following two sections, we look at the performance of TrustCEP against on-off attacks when we assume either a conservative or a liberal set of values for the coefficients.



(a) Minority Adversary Population: 20 Users, 12 Benign, 8 Adversary

(b) Majority Adversary Population: 20 Users, 6 Benign, 14 Adversary

Figure 38: Trust Value Distribution upon On-Off Attack for Conservative Trust Modification Coefficients

A.2.1 *Influence of Trust Modification Coefficients: Conservative Case*

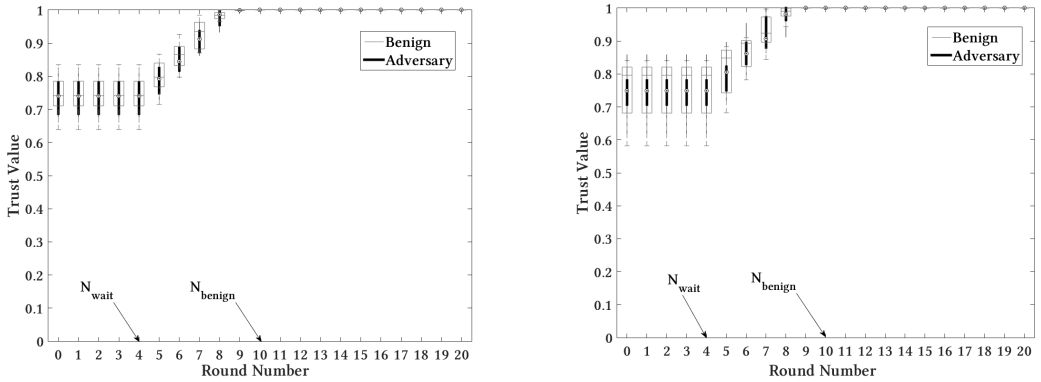
For the conservative case, as mentioned in Section 5.3.1.4, we lower the values of the trust modification coefficients, such that $\tau_{inc} = 0.025$ and $\tau_{dec} = 0.5$. In turn, any negative trust divergence[1] leads to a minor increase in the corresponding trust value. However, a positive trust divergence halves the corresponding trust value.

Figure 38 shows the trust value distribution for two samples cases, one with the adversaries in the minority, and the other with them in the majority. In Figure 38a, we observe that the conservative nature of the trust modification coefficients lead to a drastic decrease in the trust values of both the benign users as well as the adversaries, as soon as the waiting period $n_{wait} = 4$ elapses. Interestingly, even in Figure 38b, where the adversaries are in the majority, we observe a similar phenomenon, such that all users obtain a trust value of less than 0.1 by the end of 9 rounds. Although we notice that the benign users obtain higher trust values in the first rounds after the waiting period, we can see that their trust value decreases rapidly afterwards. The main reason for the poor performance of TrustCEP in the conservative case is due to the partial coverage on the part of all users in the on-off case. Recall from Section 5.3.1.3 that the adversaries only infiltrate a partial set of benign users in this case. This also affects the trust values obtained by the benign users due to the discrepancy in the trust recommendations obtained by each user. Furthermore, recall that the malicious threshold *malValue* is dependent on the trust increase coefficient τ_{inc}. Accordingly, the lower value of *malValue* implies that fewer aberrations are condoned by the users, leading to quicker punishments and therefore, lower trust values.

1 Note that a negative trust divergence indicates that the recommended trust values are higher than the current trust values. See Equation 6 in Section 5.2.2.1 for more details.

A.2.2 *Influence of Trust Modification Coefficients: Liberal Case*

For the liberal case, we used $\tau_{inc} = 0.1$ and $\tau_{dec} = 0.9$, such that the trust values obtain a large additive increase if the trust divergence with respect to the trust recommendations is negative. Similarly, a positive trust divergence leads to minimal decrease in the trust values.



(a) Minority Adversary Population: 20 Users, 12 Benign, 8 Adversary

(b) Majority Adversary Population: 20 Users, 6 Benign, 14 Adversary

Figure 39: Trust Value Distribution upon On-Off Attack for Liberal Trust Modification Coefficients

We notice in Figure 39 that the phenomenon observed in the conservative case is reversed in the liberal case. In both the sample cases for adversary minority (Figure 39a) and adversary majority (Figure 39b), we see that all users obtain a trust value of 1.0 by the end of 10 rounds. This attributes to the higher value for the malicious threshold *malValue*, as a consequence of the higher value for τ_{inc}. Subsequently, a larger number of aberrations need to occur before the corresponding users are assumed to be adversaries. This, together with the fact that the adversaries do not infiltrate the entire environment, leads to the rapid increase of the trust values of all users in the liberal case. Effectively, it is of paramount importance to choose the right trust modification coefficients for a given application.

A.3 LIST OF ACRONYMS

5G	Fifth Generation
ABAC	Attribute-Based Access Control
API	Application Programming Interface
AR	Active Replication
CBPS	Content-Based Publish/Subscribe
CEP	Complex Event Processing
D2D	Device-to-Device
DoS	Denial-of-Service
DSMS	Data Stream Management System
DTN	Delay-Tolerant Network
ECA	Event-Condition-Action
GPS	Global Positioning System
IAR	Improved Active Replication
IFP	Information Flow Processing
IM	Instant Messaging
IoT	Internet of Things
JVM	Java Virtual Machine
MANET	Mobile Ad-hoc Network
MCEP	Mobile Complex Event Processing
MOM	Message-Oriented Middleware
MSA	Mobile Situation Awareness
NB	Näive Bayes
NB-IoT	Narrowband Internet of Things
NDN	Named-Data Networking
NFC	Near Field Communication
NLP	Natural Language Processing
OS	Operating System
OSN	Online Social Network
P2P	Peer-to-Peer
PKI	Public Key Infrastructure
PP-CEP	Privacy-Preserving Complex Event Processing
RBAC	Roll-Based Access Control
RF	Random Forest
RFID	Radio Frequency Identification
RR	Rollback Recovery
SGX	Software Guard eXtensions
SMS	Short Message Service

SPS	Stream Processing System
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TTP	Trusted Third Party
UDP	User Datagram Protocol
V2V	Vehicle-to-Vehicle
VANET	Vehicular Ad-hoc Network
VM	Virtual Machine
WiFi	Wireless Fidelity
WSN	Wireless Sensor Network

A.4 SUPERVISED STUDENT THESES

A.4.1 *Bachelor and Master Theses*

[1] Yashas Bharadwaj. "A Smartphone-Based Distributed System for Privacy-Preserving Complex Event Processing in Dynamic User Environments." Master Thesis. TU Darmstadt, 2016.

[2] Jérôme Charrier. "Design of a Familiarity Function to Preserve User Privacy in Context Data Exchange." Bachelor Thesis. TU Darmstadt, 2015.

[3] Christof Pauer. "Vertraulicher Informationsaustausch in mobilen kontextbewussten Anwendungen mittels des Publish/Subscribe Paradigmas." Engl.: "Confidential Information Exchange for Mobile Context-aware Applications using the Publish/Subscribe Paradigm". Master Thesis. TU Darmstadt, 2016.

[4] Johannes Thomas. "Entwurf einer hybriden Architektur zum Austausch von sensiblen Daten in mobilen kontextsensitiven Anwendungen durch geeignetes Caching." Engl.: "Design of a Hybrid Architecture for Privacy-Preserving Data Exchange in Mobile Context-aware Applications through Suitable Caching". Master Thesis. TU Darmstadt, 2016.

[5] Thulasiram Valleru. "Efficient Migration of Operator Graphs for Distributed Complex Event Processing in Dynamic User Environments." Master Thesis. TU Darmstadt, 2017.

A.4.2 *Labs and Seminars*

[6] Abhishek Ballal and Bharath Honnappa. "Information Confidentiality in Event Processing Systems." Seminar on Advanced Topics in Future Internet Research. TU Darmstadt, 2017.

[7] Fares Beji. "Adaptive Video Streaming in Wireless Named Data Networking." Lab Project. TU Darmstadt, 2016.

[8] Jérôme Charrier, Shule Liu, and Shahnewaz Suman. "Analyzing Familarity based on User Interactions." Lab Project. TU Darmstadt, 2016.

[9] Muhammad Talha Khan. "Privacy Preservation during Information Exchange over the Internet." Seminar on Advanced Topics in Future Internet Research. TU Darmstadt, 2015.

[10] Seema Kumar and R. K. Dileep. "Secure Range Queries in Wireless Sensor Networks." Seminar on Advanced Topics in Future Internet Research. TU Darmstadt, 2015.

[11] Artur Michel. "Survey über Lokalisierungs- und Kontextuierungsverfahren von Nutzern." Engl.: "Survey on Approaches for Localization and Contextualization of Users". Proseminar. TU Darmstadt, 2015.

[12] Artur Michel and Long Zhao. "On-the-Fly Decentralized Mechanisms to Identify Co-located Peers." Lab Project. TU Darmstadt, 2015.

[13] Nivedita Paul and Nikhil Kalathil. "An Analysis of Privacy-Preserving Mechanisms in Distributed Systems." Seminar on Advanced Topics in Future Internet Research. TU Darmstadt, 2015.

[14] Daniel Specht, Benjamin Schiller, and Marius Rettberg-Päplow. "Activity Recognition Based on User Interaction Patterns on Smartphones." Lab Project. TU Darmstadt, 2017.

[15] Navdeep Uniyal and Vamsi Krishna Sripathi. "Secure Range Queries in Sensor Networks." Seminar on Advanced Topics in Future Internet Research. TU Darmstadt, 2015.

B

AUTHOR'S PUBLICATIONS

MAIN PUBLICATIONS

[1] Rahul Dwarakanath, Jérôme Charrier, Frank Englert, Ronny Hans, Dominik Stingl, and Ralf Steinmetz. "Analyzing the Influence of Instant Messaging on User Relationship Estimation." In: *2016 IEEE International Conference on Mobile Services (MS)*. San Francisco, CA, USA: IEEE, 2016, pp. 49–56.

[2] Rahul Dwarakanath, Boris Koldehofe, Yashas Bharadwaj, The An Binh Nguyen, David Eyers, and Ralf Steinmetz. "TrustCEP: Adopting a Trust-Based Approach for Distributed Complex Event Processing." In: *2017 IEEE International Conference on Mobile Data Management (MDM)*. Daejeon, South Korea: IEEE, 2017, pp. 30–39.

[3] Rahul Dwarakanath, Boris Koldehofe, and Ralf Steinmetz. "Operator Migration for Distributed Complex Event Processing in Device-to-Device Based Networks." In: *Proceedings of the 3rd Workshop on Middleware for Context-Aware Applications in the IoT*. Trento, Italy: ACM, 2016, pp. 13–18.

[4] Rahul Dwarakanath, Dominik Stingl, and Ralf Steinmetz. "Improving Inter-user Communication: A Technical Survey on Context-aware Communication." In: *PIK-Praxis der Informationsverarbeitung und Kommunikation* 38.1-2 (2015).

CO-AUTHORED PUBLICATIONS

[5] Svenja Neitzel, Frank Englert, Rahul Dwarakanath, Katharina Schneider, Kathrin Reinke, Gisela Gerlach, Christoph Rensing, Doreen Boehnstedt, and Ruth Stock-Homburg. "Towards Using Situational Information to Detect an Individual's Perceived Stress Level." In: *2017 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. Kona, HI, USA: IEEE, 2017, pp. 333–338.

[6] The An Binh Nguyen, Pratyush Agnihotri, Christian Meurisch, Manisha Luthra, Rahul Dwarakanath, Jeremias Blendin, Doreen Boehnstedt, Michael Zink, and Ralf Steinmetz. "Efficient Crowd Sensing Task Distribution Through Context-aware NDN-based Geocast." In: *42nd IEEE International Conference on Local Computer Networks (LCN)*. Singapore, Singapore: IEEE, 2017, pp. 52–60.

[7] Katharina Schneider, Frank Englert, Doreen Böhnstedt, Gisela Bieling, Rahul Dwarakanath, Ralf Steinmetz, and Ruth Stock-Homburg. "Why are You so Stressed? Analyzing Methods for Mobile Stress Measurement." In: *Digital Working Life*. Dubrovnik, Croatia, 2016.

[8] Katharina Schneider, Kathrin Reinke, Gisela Gerlach, Christoph Anderson, Sebastian Wojtek, Svenja Neitzel, Rahul Dwarakanath, Doreen Boehnstedt, and Ruth Stock. "Aligning ICT-enabled Availability and Individual Availability Preferences: Design and Evaluation of Availability Management Applications." In: *Proceedings of International Conference on Information Systems (ICIS)*. Seoul, South Korea: AIS Electronic Library, 2017.

DEMO/PHD FORUM PAPERS

[9] Rahul Dwarakanath and Ralf Steinmetz. "A Decentralized System for Privacy-Preserving Context Exchange: Facilitating a Better Work-Life Balance." In: *2014 IEEE 22nd International Conference on Network Protocols (ICNP): PhD Forum*. 2014, pp. 489–491.

[10] Denny Stohr, Fares Beji, Rahul Dwarakanath, Ralf Steinmetz, and Wolfgang Effelsberg. "Mobile NDN-Based Dynamic Adaptive Streaming over HTTP." In: *41st IEEE International Conference on Local Computer Networks (LCN): Demo Track*. Dubai, UAE: IEEE, 2016, pp. 1–4.

C

CURRICULUM VITÆ

PERSONAL

Name	Rahul Wermund, ne Chini Dwarakanath
Date of Birth	September 05, 1987
Place of Birth	Raichur, Karnataka, India
Nationality	Indian

EDUCATION

Since 01/2014	Technische Universität Darmstadt, Doctoral candidate at the Department of Electrical Engineering and Information Technology
10/2010-07/2013	Technische Universität Darmstadt, Master studies in Electrical Engineering, Degree: Master of Science
09/2005-06/2009	Visweswaraya Technological University, India Bachelor studies in Telecommunications, Degree: Bachelor of Engineering

ACADEMIC EXPERIENCE

01/2014-12/2017	Researcher funded by the Social Link Project within the Loewe Program of Excellence in Research, Hesse, Germany

WORK EXPERIENCE

01/2014-12/2017	Technische Universität Darmstadt, Research assistant in the research group Adaptive Overlay Communications (AOC) at the Multimedia Communcations Lab (KOM)
06/2013-07/2013	Nomor Research GmbH, Student trainee (DE: Werkstudent) in the System Level Simulation group

04/2012-09/2012	Nomor Research GmbH, Intern in the System Level Simulation group

TEACHING ACTIVITY

2014 - 2017	Lecture "Communication Networks II", Organizer and teaching assistant.
2014 - 2017	Lab "Multimedia Communications Lab/Project", Supervisor.
2014 - 2017	Seminar "Advanced Topics in Future Internet Research", Supervisor.
2014	Seminar "Mobile Phone-based Participatory Sensing and its Analysis in Mobile Communities", Teaching assistant

SCIENTIFIC ACTIVITIES

2017	Reviewer for ACM International Conference on Distributed and Event-Based Systems (DEBS), 2017
2017	Reviewer for International Conference on Cloud Computing and Services Science (CLOSER), 2017 (on behalf of Prof. Dr.-Ing. Ralf Steinmetz)
2016	Reviewer for ACM/IEEE International Conference on Internet-of-Things Design and Implementation (IoTDI), 2016 (on behalf of Prof. Dr.-Ing. Ralf Steinmetz)
2016	Reviewer for ACM International Conference on Distributed and Event-Based Systems (DEBS), 2016 (on behalf of Dr. Boris Koldehofe)
2014	Reviewer for IEEE International Conference on Peer-to-Peer Computing (P2P), 2014 (on behalf of Prof. Dr.-Ing. Ralf Steinmetz)

SUPERVISED THESES

01/2017	Thulasiram Valleru, "Efficient Migration of Operator Graphs for Distributed Complex Event Processing in Dynamic User Environments", Master thesis
08/2016	Yashas Bharadwaj, "A Smartphone-Based Distributed System for Dynamic Privacy-Preserving Complex Event Processing", Master thesis

07/2016	Johannes Thomas, "Entwurf einer hybriden Architektur zum Austausch von sensiblen Daten in mobilen kontextsensitiven Anwendungen durch geeignetes Caching", Master thesis
03/2016	Christof Pauer, "Vertraulicher Informationsaustausch in mobilen kontextbewussten Anwendungen mittels des Publish/-Subscribe Paradigmas", Master thesis
09/2015	Jérôme Charrier, "Design of a Familiarity Function to Preserve User Privacy in Context Data Exchange", Bachelor thesis

HONORS

2005-2009	'Distinction award' by Visweswaraya Technological University for securing First Class with Distinction (FCD) through all 8 semesters of the Bachelor studies

Darmstadt, January 23, 2018

Rahul Wermund

D

ERKLÄRUNG LAUT §9 DER PROMOTIONSORDNUNG

Ich versichere hiermit, dass ich die vorliegende Dissertation allein und nur unter Verwendung der angegebenen Literatur verfasst habe. Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

Darmstadt, 30. Oktober 2017

Rahul Wermund, geb. Chini Dwarakanath