CONTEXT AND COMMUNICATION PROFILING FOR IOT SECURITY AND PRIVACY: TECHNIQUES AND APPLICATIONS

Vom Fachbereich Informatik
der Technischen Universität Darmstadt

zur Erlangung des Grades
Doktor-Ingenieur (Dr.-Ing.)
genehmigte

Dissertation
von M.Sc. Markus Miettinen
aus Helsinki, Finnland

Erstgutachter: Prof. Dr.-Ing. Ahmad-Reza Sadeghi
Zweitgutachter: Prof. N. Asokan, PhD

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Darmstadt 2018

Markus Miettinen: *Context and communication profiling for IoT security and privacy: techniques and applications*
Darmstadt, Technische Universität Darmstadt
Jahr der Veröffentlichung der Dissertation auf TUprints: 2019
URN: urn:nbn:de:tuda-tuprints-83742
Tag der mündlichen Prüfung: 12.12.2018
Version 1.0

Omistettu Tuulalle, jota ilman tämän väitöskirjan laatiminen ja siihen liittyvä tutkimustyö eivät olisi olleet mahdollisia.

ABSTRACT

During the last decade, two major technological changes have profoundly changed the way in which users consume and interact with on-line services and applications. The first of these has been the success of mobile computing, in particular that of smartphones, the primary end device used by many users for access to the Internet and various applications. The other change is the emergence of the so-called Internet of Things (IoT), denoting a technological transition in which everyday objects like household appliances that traditionally have been seen as stand-alone devices, are given network connectivity by introducing digital communication capabilities to those devices. The topic of this dissertation is related to a core challenge that the emergence of these technologies is introducing: how to effectively manage the security and privacy settings of users and devices in a user-friendly manner in an environment in which an ever-growing number of heterogeneous devices live and co-exist with each other?

In particular we study approaches for utilising profiling of contextual parameters and device communications in order to make autonomous security decisions with the goal of striking a better balance between a system's security on one hand, and, its usability on the other. We introduce four distinct novel approaches utilising profiling for this end. First, we introduce ConXsense, a system demonstrating the use of user-specific longitudinal profiling of contextual information for modelling the usage context of mobile computing devices. Based on this ConXsense can probabilistically automate security policy decisions affecting security settings of the device. Further we develop an approach utilising the similarity of contextual parameters observed with on-board sensors of co-located devices to construct *proofs of presence* that are resilient to context-guessing attacks by adversaries that seek to fool a device into believing the adversary is co-located with it, even though it is in reality not. We then extend this approach to a *context-based key evolution approach* that allows IoT devices that are co-present in the same physical environment like the same room to use passively observed context measurements to *iteratively authenticate their co-presence* and thus gradually establish confidence in the other device being part of the same *trust domain*, e.g., the set of IoT devices in a user's home. We further analyse the relevant constraints that need to be taken into account to ensure security and usability of context-based authentication. In the final part of this dissertation we extend the profiling approach to network communications of IoT devices and utilise it to realise the design of the IoTSentinel system for autonomous security policy adaptation in IoT device net-

works. We show that by monitoring the inherent network traffic of IoT devices during their initial set-up, we can *automatically identify the type* of device newly added to the network. The device-type information is then used by IoTSentinel to adapt traffic filtering rules automatically to provide isolation of devices that are potentially vulnerable to known attacks, thereby protecting the device itself and the rest of the network from threats arising from possible compromise of vulnerable devices.

ZUSAMMENFASSUNG

In den letzten zehn Jahren haben zwei wichtige technologische Veränderungen die Art mit der digitale Onlineangebote konsumiert und Onlineservices benutzt werden, grundlegend verändert. Zum einen hat der große Erfolg von mobilen Endgeräten, insbesondere der von Smartphones dazu geführt, dass mittlerweile mobile Geräte für viele Nutzer das hauptsächliche Medium sind, mit dem sie sich mit dem Internet verbinden und verschiedene Anwendungen benutzen. Die andere große Veränderung ist der anhaltende Siegeszug des sogenannten *Internets der Dinge (Internet of Things, IoT)*, einem technologischen Trend, bei dem Alltagsgegenstände im Haushalt, die typischerweise bisher nicht vernetzt waren, mit einer Netzwerkverbindung und computergesteuerten intelligenten Funktionalitäten ausgestattet werden. Das tragende Thema dieser Dissertation fokussiert sich auf die Herausforderungen, die durch das Aufkommen dieser neuen Technologietrends verursacht werden, und zwar der Frage, wie man die Sicherheits- und Privatheitseinstellungen von Anwendungen und Geräten in einer effizienten und benutzerfreundlichen Art in einer Umgebung mit immer mehr sehr unterschiedlichen Geräten handhaben kann.

Insbesondere untersuchen wir verschiedene Ansätze zur Profilierung von Kontextparametern und Kommunikationsmustern von Geräten mit deren Hilfe autonome Entscheidungen über Sicherheitseinstellungen getroffen werden können. Hierbei ist es unser Ziel, eine bessere Balance zwischen der Sicherheit des Systems einerseits, und seiner einfachen Handhabung durch Nutzer andererseits zu finden. Um auf dieses Ziel zuzuarbeiten, stellen wir vier verschiedene neue Profilierungsansätze vor. Zum einen präsentieren wir ConXsense, ein System welches demonstriert, wie nutzerspezifische Langzeitprofile von kontextuellen Informationen die von mobilen Geräten mit ihren Kontextsensoren erfasst werden, dazu benutzt werden können, um die Nutzungssituationen, in denen mobile Endgeräte eingesetzt werden, zu modellieren. Darauf basierend kann ConXsense das Treffen von Entscheidungen über sicherheits- und privatheitsrelevante Einstellungen mithilfe eines probabilistischen Vorhersagemodells automatisieren. Weiterhin entwickeln wir einen Ansatz, der die Ähnlichkeit von kontextuellen Parametern, die Geräte die in unmittelbarer Nähe voneinander platziert sind, mit ihren Kontextsensoren erfassen, ausnutzt, um einen *Präsenzbeweis (Proof-of-Presence, PoP)* zu erstellen. Zudem zeigen wir, wie diese Präsenzbeweise erstellt und ausgewertet werden müssen, um widerstandsfähig gegen Angriffe zu sein, in denen ein Angreifer versucht, durch verfälschte Präsenzbeweise ein Gerät zu dem Glauben zu verleiten, dass der Angreifer in der un-

mittelbaren Nähe des Geräts ist, obwohl dies in Wirklichkeit nicht der Fall ist. Im Anschluss erweitern wir diesen Ansatz in einen *kontextbasierten Schlüsselevolutionsansatz*, der es IoT-Geräten, die in der selben physischen Umgebung wie z. B. im selben Zimmer platziert sind, erlaubt, ihre passiv erfassten Kontextmessungen zu benutzen, um *iterativ die Präsenz des anderen Gerätes in der selben Umgebung zu authentifizieren*. Hierdurch können Geräte nach und nach die Zugehörigkeit des anderen Gerätes in der selben administrativen Gruppe von Geräten, z. B. der Gruppe der IoT-Geräte im Smart Home eines Nutzers, authentifizieren. Hinsichtlich dieses kontextbasierten Authentifizierungsansatzes analysieren wir relevante Faktoren und Randbedingungen die berücksichtigt werden müssen, um die Sicherheit und Benutzerfreundlichkeit des Ansatzes sicherzustellen.

Im letzten Abschnitt dieser Dissertation erweitern wir unseren Profilierungsansatz auf die Netzwerkkommunikationen von IoT-Geräten. Wir benutzen dies um das Design von IoTSentinel, einem System für die autonome Verwaltung von Sicherheitseinstellungen in IoT-Netzwerken zu realisieren. Im Rahmen dieser Arbeit können wir zeigen, dass die charakteristischen Kommunikationsmuster von IoT-Geräten während ihrer erstmaligen Installation im Netzwerk dazu benutzt werden können, um den *Typ eines IoT-Gerätes automatisch zu identifizieren*, sobald es in das Netzwerk hinzugefügt wird. Diese Information wird wiederum von IoTSentinel dazu benutzt, um automatisch Regeln zum Filtern von Netzwerkkommunikationen zu erstellen, sodass Gerätetypen, die bekannte Sicherheitsmängel aufweisen und deshalb potenziell für Angriffe anfällig sind, von anderen Geräten im Netzwerk isoliert werden können. Dies dient zum einen dazu, Angriffen vorzubeugen, die versuchen das betroffene Gerät zu kompromittieren, und zum anderen auch dazu, den Rest des Netzwerks vor möglichen Gefährdungen durch möglicherweise erfolgreich infizierte Geräte zu schützen.

TIIVISTELMÄ

Kaksi merkittävää teknologista murrosta ovat kuluneen vuosikymmenen aikana muuttaneet merkittävästi tapaa, jolla käyttäjät kuluttavat verkkopalveluita ja käyttävät verkossa olevia järjestelmiä. Mobiilin tietojenkäsittelyn, erityisesti älypuhelimien suuri menestys on johtanut siihen, että yhä useampi käyttäjä käyttää internetiä ja erilaisia sovelluksia pääasiassa älypuhelimen avulla. Toinen merkittävä uudistus liittyy *esineiden internetin (Internet of Things, IoT)* yleistymiseen. Tällä tarkoitetaan arkisten laitteiden kuten kodinkoneiden, jotka perinteisesti eivät ole olleet yhdistetty verkkoon, varustamista verkkoyhteydellä ja tietokoneohjatulla älykäällä toiminnallisuudella. Tämän väitöskirjan aihe keskittyy erääseen keskeisimmistä haasteista, joka näihin teknologisiin murroksiin liittyy: kysymykseen siitä, kuinka hallita käyttäjien ja laitteiden turva- ja yksityisyysasetuksia ympäristössä, jossa on yhä enemmän erilaisia ja erittäin monipuolisia toistensa kanssa vuorovaikutuksessa olevia laitteita.

Tässä työssä tutkimme erityisesti lähestymistapoja, jotka hyödyntävät laitteiden ympäristöstään havainnoimien kontekstiparametrien ja tietoliikenteen profilointia autonomisten tietoturvapäätösten tekemisessä. Tavoitteenamme on löytää parempi tasapaino toisaalta järjestelmän tietoturvan tason ja toisaalta järjestelmän käytettävyyden ja käyttäjäystävällisyyden välillä. Tämän saavuttamiseksi esittelemme neljä erilaista profilointiin perustuvaa menetelmää. Esittelemme ConXsense-nimisen järjestelmän, joka demonstroi pitkäaikaisen käyttäjäkohtaisen kontekstitiedon profiloinnin käyttöä mobiililaitteen käyttökontekstin mallintamisessa. Tämän mallinnuksen perusteella ConXsense-järjestelmä kykenee probabilistisesti tekemään automaattisia tietoturvapolitiikkoihin liittyviä päätöksiä, jotka vaikuttavat mobiililaitteen tietoturva- ja yksityisyysasetuksiin. Seuraavaksi kehitämme kontekstin profilointiin perustuvan menetelmän, joka hyödyntää samassa tilassa sijaitsevien laitteiden kontekstisensoreillaan tekemien mittausten samankaltaisuutta kontekstiperustaisten *läsnäolotodisteiden (Proof-of-Presence)* toteuttamisessa. Osoitamme, kuinka nämä läsnäolotodisteet pitää laatia, jotta ne olisivat vastustuskykyisiä sellaisia hyökkäyksiä vastaan, joissa hyökkääjä pyrkii harhauttamaan laitetta uskomaan hyökkääjän olevan läsnä samassa tilassa laitteen kanssa, vaikka tämä on tosiasiassa muualla. Tämän jälkeen sovellamme samankaltaista kontekstin profilointimenetelmää *kontekstiperustaisessa todennusavainten kehitysmenetelmässä (key evolution approach)*, jonka avulla samassa fyysisessä tilassa, kuten esimerkiksi samassa huoneessa, sijaitsevat IoT-laitteet voivat iteratiivisesti todentaa toistensa läsnäolon samassa tilassa hyödyntämällä passiivisesti havainnoituja kontekstiparametreja. Näin menettelemällä laitteet voivat asteittain kasvattaa luottamus-

taan siihen, että toinen laite sijaitsee samassa tilassa ja kuuluu siten samaan luotettujen laitteiden ryhmään, kuten esimerkiksi käyttäjän kotona sijaitsevien IoT-laitteiden muodostamaan ryhmään. Esittelemäämme menetelmään liittyen analysoimme relevantteja rajoitteita, jotka pitää ottaa huomioon varmistaaksemme saavutettavan todennuksen riittävän vastustuskyvyn hyökkäyksiä vastaan sekä käytettävyyden käyttäjien kannalta. Väitöskirjan viimeisessä osiossa laajennamme profilointimenetelmien käytön IoT-laitteiden laitekohtaiseen tietoliikenteeseen. Suunnittelemme IoTSentinel-järjestelmän, jonka tarkoituksena on mahdollistaa IoT-laitteisiin liittyvien verkkoasetusten automaattinen mukauttaminen lähiverkoissa. Esitämme kuinka tarkkailemalla ja profiloimalla IoT-laitteen tietoliikennettä sitä verkkoon asennettaessa voimme automaattisesti tunnistaa asennettavan laitteen tyypin. IoTSentinel-järjestelmä hyödyntää tietoa laitteen tyypistä mukauttaakseen automaattisesti laitteeseen sovellettavia tietoliikenteen suodatussääntöjä. Sääntöjen avulla järjestelmä kykenee muun muassa eristämään sellaiset laitteet, joiden tiedetään olevan haavoittuvaisia tietoturvahyökkäyksille. Näin menettelemällä pyritään toisaalta suojaamaan haavoittuvaista laitetta laitteen haavoittuvaisuutta hyödyntämään pyrkiviltä tietoturvahyökkäyksiltä ja toisaalta suojaamaan myös muita verkossa sijaitsevia laitteita siinä tapauksessa, että hyökkääjä onnistuu menestyksekkäästi hyökkäämään haavoittuvaista laitetta vastaan ja kaappaamaan sen hallintaansa käyttääkseen sitä hyökkäyksiin muita laitteita vastaan.

ACKNOWLEDGEMENTS – DANKSAGUNG – KIITOKSET

I would like to thank my advisor Prof. Ahmad-Reza Sadeghi for his support in pursuing the research for this thesis and providing an excellent opportunity to work in a team of highly talented researchers in the security field. During the past years I have yhad the possibility to learn a lot about the practice of security research at the highest level.

I would also like to express my warmest gratitude to my co-advisor Prof. Asokan, who has been a reliable mentor and source for enthusiasm for a long time even before I had the opportunity to join academia. Prof. Asokan's advice and generous support have been an invaluable resource for me during most of my researcher career.

Warm thanks also to all my fellow co-authors Thien Duc Nguyen, Dr. Samuel Marchal, Dr.-Ing. Stephan Heuser, Wiebke Kronz, Majid Sobhani, Tommaso Frassetto, Ibbad Hafeez, Jon Rios and Sudha Yellapantula with whom I have had the privilege to work on interesting research questions and discuss particular issues and technical problems. I would also extend my thanks to Prof. Farinaz Koushanfar and Prof. Sasu Tarkoma for their contributions to the research work underlying this dissertation.

A special thanks goes also to all the members of the System Security Lab at Technische Universität Darmstadt with which I have had the pleasure to work during the recent years.

Ein warmes Dankeschön geht auch an alle Freunde und Bekannte, die mich während meiner Laufbahn mit Rat und Tat unterstützt und ermutigt haben. Ohne das Umfeld guter Freunde und positiver Teilhabe an meinem Leben hätte ich wohl nicht die Kraft gehabt, mich den Herausforderungen der Wissenschaft und des Lebens im weiteren Sinne zu stellen, die für das Durchführen der Forschungsarbeit für diese Dissertation erforderlich waren.

Sydämelliset kiitokseni kaikille ystäville, kollegoille ja yhteistyökumppaneille, joiden tuen ja neuvojen avulla minulla on riittänyt uskoa ja mielenkiintoa omistautua tämän väitöskirjan vaatimalle tutkimustyölle ja ratkoa siihen liittyviä tieteellisiä ja toisanaan hyvinkin käytännöllisiä haasteita. Aivan erityisen kiitoksen haluan myös lausua vanhemmilleni, isovanhemmilleni sekä veljilleni ja sukulaisilleni, joiden ansiosta olen aina saanut nauttia luottavaisesta ja kannustavasta ilmapiiristä, jonka pohjalta on ollut helppo tarttua uusiin haasteisiin ja heittäytyä elämän vietäväksi niin kotona kuin kaukana ulkomaillakin. Ilman perhepiiristä saatua luottamusta ja positiivista

elämänasennetta olisi moni projekti ja ajatus jäänyt syntymättä ja niinollen tähän väitöskirjaankin johtanut elämänpolku kulkematta.

Lopuksi haluan kiittää perhettäni, vaimoani Tuulaa sekä poikiamme Anttia ja Eeroa. Ilman teidän kannustustanne ja tukeanne sekä ymmärrystänne isän ja miehen välillä ehkä liiankin syvälliselle uppoutumiselle työhön ei tämä tutkimus olisi ollut mahdollista toteuttaa. Olen hyvin kiitollinen ja iloinen siitä, että minulla on ollut onni ja ilo elää kanssanne koko tämän pitkän prosessin ajan.

CONTENTS

1 INTRODUCTION 1
1.1 Challenges . . . 2
1.1.1 Context-Aware Access Control . . . 2
1.1.2 Context-Based Proofs-of-Presence . . . 2
1.1.3 Context-Based Authentication . . . 3
1.1.4 Security Management of IoT Devices . . . 4
1.2 The Goal of this Dissertation . . . 4
1.3 Overview of Contributions . . . 5
1.3.1 Context-Aware Access Control Framework . . . 5
1.3.2 Resilient Context-Based Proofs-of-Presence . . . 5
1.3.3 Context-Based Authentication . . . 5
1.3.4 Device Profiling for IoT Security Management . 6
1.4 Related Publications . . . 6
1.5 Contributions of the Author . . . 7
1.5.1 Context-Aware Access Control Framework . . . 7
1.5.2 Resilient Context-Based Proofs-of-Presence . . . 8
1.5.3 Context-Based Authentication . . . 8
1.5.4 Device Profiling for IoT Security Management . 9

2 CONTEXT PROFILING FOR AUTOMATING SECURITY AND PRIVACY POLICY DECISIONS 10
2.1 Problem Description . . . 10
2.1.1 Challenges with Straightforward Approaches . 11
2.1.2 Goal Setting . . . 12
2.1.3 Definition of the Term Context . . . 12
2.1.4 User Perceptions of Context . . . 12
2.1.5 Use Cases . . . 13
2.1.6 Adversary Model . . . 14
2.2 System Design . . . 15
2.3 Context Model . . . 16
2.3.1 Profiling Locations and Places . . . 16
2.3.2 Profiling the Social Context . . . 21
2.3.3 Context Features . . . 22
2.4 Evaluation . . . 22
2.4.1 Data Collection . . . 22
2.4.2 Dataset . . . 25
2.4.3 Context Classification . . . 25
2.5 Enforcement . . . 28
2.6 Related Work . . . 29
2.6.1 Use of Contextual Data for User Profiling . . . 29
2.6.2 Context-Aware Access Control . . . 29

2.6.3 Context-Based Access Control Enforcement in Mobile Systems . . . 30
2.6.4 Usable Access Control . . . 31
2.6.5 Context Identification . . . 32
2.6.6 Context-Aware Policy Adaptation . . . 33
2.7 Summary and Conclusions . . . 33

3 CONTEXT PROFILING FOR RESILIENT PROOFS OF PRESENCE 36
3.1 Problem Description . . . 36
3.1.1 Proofs-of-Presence . . . 37
3.1.2 Context Guessing Attacks . . . 38
3.1.3 Goals and Contributions . . . 39
3.2 Background . . . 39
3.2.1 Use of WiFi for Proximity Verification . . . 39
3.2.2 Multi-Modal Proximity Verification . . . 40
3.3 Context-Based Proofs-of-Presence . . . 41
3.3.1 Context Features . . . 41
3.4 Context Guessing Attacks . . . 42
3.4.1 Adversary Model . . . 43
3.4.2 Context Guessing . . . 43
3.4.3 Susceptibility of PoPs to Context-Guessing Attacks . . . 44
3.5 Hardening Context-Based Proofs-of-Presence . . . 48
3.5.1 Surprisal Filtering . . . 49
3.5.2 Longitudinal Ambient Modalities . . . 51
3.6 Evaluation . . . 53
3.6.1 Performance of Surprisal Filtering . . . 53
3.7 Discussion . . . 55
3.7.1 Impact of Context Entropy . . . 55
3.7.2 Privacy Considerations . . . 56
3.8 Related Work . . . 57
3.8.1 Beaconing-Based approaches . . . 57
3.8.2 Proofs-of-Presence Based on Context . . . 58
3.8.3 Distance-Bounding Approaches . . . 59
3.9 Summary and Conclusion . . . 59

4 CONTEXT-BASED AUTHENTICATION OF IOT DEVICES 61
4.1 Background . . . 62
4.1.1 Key Pre-Sharing . . . 62
4.1.2 Demonstrative Authentication . . . 63
4.1.3 Context-Based Pairing Approaches . . . 64
4.1.4 Problem Description . . . 66
4.1.5 IoT Device Pairing Scenario . . . 66
4.1.6 Wearable Device On-Boarding Scenario . . . 67
4.2 Adversary Model . . . 67
4.2.1 Adversaries in IoT Pairing . . . 67
4.2.2 Adversaries in Wearable Device On-Boarding . . . 68

4.3 System Design . . . 69
4.3.1 Goals and Requirements . . . 69
4.3.2 Solution Intuition . . . 70
4.3.3 Context-Based Key Evolution . . . 70
4.4 Context Fingerprinting . . . 75
4.4.1 Context Measurements . . . 76
4.4.2 Context Quantisation . . . 76
4.5 Evaluation . . . 78
4.5.1 Evaluation Metrics . . . 78
4.5.2 Experiment Set-Up . . . 79
4.5.3 Datasets . . . 81
4.5.4 Fingerprint Similarity . . . 82
4.5.5 Fingerprint Entropy . . . 85
4.6 Security Analysis . . . 88
4.6.1 Entropy Analysis . . . 89
4.6.2 Authentication Performance . . . 92
4.7 Related Work . . . 94
4.7.1 Key Pre-Sharing Schemes . . . 94
4.7.2 Context-Based Schemes . . . 97
4.8 Summary and Conclusion . . . 99

5 SECURITY MANAGEMENT IN IOT BASED ON DEVICE PROFILING 101
5.1 Problem Description . . . 102
5.1.1 Insufficience of Software Patching . . . 102
5.1.2 Need for Brownfield Solutions . . . 103
5.1.3 Goal and Contributions . . . 103
5.2 Adversary Model . . . 104
5.2.1 Adversary Goals . . . 104
5.2.2 Assumptions . . . 105
5.3 System Design . . . 105
5.3.1 System Components . . . 106
5.3.2 Device Fingerprinting . . . 107
5.3.3 Device-Type Identification . . . 108
5.3.4 Vulnerability Assessment . . . 109
5.3.5 Enforcement . . . 110
5.4 Evaluation . . . 112
5.4.1 Experiment Set-Up . . . 112
5.4.2 Prototype Implementation . . . 116
5.5 Security Analysis . . . 120
5.5.1 Masquerading Adversary . . . 120
5.5.2 Impact of Device Mis-Classification . . . 123
5.5.3 Attacks Against IoTSentinel Components . . . 124
5.5.4 Data Poisoning Attacks . . . 125
5.6 Related Work . . . 128
5.6.1 Device Fingerprinting . . . 128
5.6.2 Device-Type Fingerprinting . . . 130

5.6.3 Device Authentication in IoT 131
5.6.4 Run-Time Security Enforcement 132
5.6.5 Commercial IoT Security Solutions 132
5.7 Summary and Conclusion 133

6 DISCUSSION AND CONCLUSION 135
6.1 Summary of Dissertation 135
6.1.1 Context Profiling for Policy Adaptation 135
6.1.2 Context-Based Proofs-of-Presence 136
6.1.3 Context-Based Authentication 136
6.1.4 IoT Security Management Based on Communications Profiling 137
6.2 Future Research Directions 138
6.2.1 Communicating Inferred Security and Privacy Settings to Users 138
6.2.2 Practical Considerations for Context-Based Authentication 138
6.2.3 Extension of Device-Type Identification to On-Line Device Behaviour 139
6.2.4 Enhanced Notions of Device Types 139
6.2.5 Augmenting Proactive with Reactive Defences . 139

7 ABOUT THE AUTHOR 140

BIBLIOGRAPHY 145

LIST OF FIGURES

Figure 2.1 The structure of the ConXsense system 16
Figure 2.2 GUI of the Context Collector app used for collecting ground truth information about the risk of device misuse and privacy sensitivity of the context. 'Safe' indicates a context with low risk of device misuse, whereas 'Unsafe' indicates high risk. 'Private' and 'Work' indicate contexts with high privacy sensitivity, and 'Public' indicates a context not considered sensitive from a privacy point of view. 24
Figure 2.3 Average ROC curves showing performance of classifying contexts with a low risk of device misuse, considering those users who provided at least five ground truth feedbacks per context class. 26
Figure 2.4 Receiver Operating Characteristic (ROC) curves of classifier performance in identifying public contexts in which sensory malware protection can be relaxed. 27
Figure 3.1 Verification of *check-ins* in location-based services (LBS) 37
Figure 3.2 Context-based proof-of-presence 42
Figure 3.3 Adversary model 43
Figure 4.1 IoT device pairing scenario 66
Figure 4.2 Context-Based Key Evolution 74
Figure 4.3 Fraction of 1-bits in and entropy of 60-bit fingerprints with window size $d = 120\,\mathrm{s}$ in the office setting of experiment 1 during different times of day 87
Figure 4.4 Bitrate of fingerprint extraction during different times of day in experiment 2 utilising peak-based list encoding 87
Figure 4.5 FAR vs. FRR for error-correction levels 5 %, 8 %, 10 %, 12 % and 15 % for different fingerprint lengths 95
Figure 4.6 Number of required authentication iterations to reach authentication strength corresponding to adversary success probability $P_{max} = 2^{-20}$. 96
Figure 5.1 The IoTSentinel adversary model 104
Figure 5.2 IoTSentinel system design 106

Figure 5.3 Security policy enforcement with IoTSentinel isolation levels . . . 111
Figure 5.4 Typical set-up process of WiFi-based wireless IoT devices . . . 113
Figure 5.5 Overview of the IoTSentinel prototype design . 117
Figure 5.6 Time required for fingerprint extraction, compression and device-type identification for different device types (1: *Fitbit Aria*, 2: *Edimax Plug 1101W*, 3: *Edimax Plug 2101W*, 4: *Ednet Gateway*, 5: *TP-Link Plug HS100*, 6: *TP-Link Plug HS110*) . . . 119

LIST OF TABLES

Table 2.1 Context features derived based on the context model . . . 23
Table 3.1 Features used for context-based PoPs . . . 42
Table 3.2 Measured context parameters in the ConXPoP dataset . . . 46
Table 3.3 Co-location classifier FPR on benign datasets . 47
Table 3.4 FPR of the co-location classifier in view of context-guessing attacks . . . 48
Table 3.5 Additional feature for luminosity and audio measurements . . . 53
Table 3.6 Improvement in FPR when surprisal filtering is applied on the ConXPoP attack datasets . . . 54
Table 3.7 Performance of PoPs utilizing audio and luminosity modalities . . . 55
Table 4.1 Set-up of IoT scenario experiments . . . 80
Table 4.2 Average fingerprint similarity between the co-located and adversary devices in the IoT scenario experiment 1 . . . 83
Table 4.3 Error-correction levels and minimum required min-entropy $\widetilde{\mathbf{H}}_\infty(W)$ of used fingerprints w to obtain $\widetilde{\mathbf{H}}_\infty(W|P) \geq 20$ bits needed for an authentication strength of $P_{max} = 2^{-20}$ in different experimental settings. The last two colums show the average entropy rates of fingerprints and the corresponding minimum bit length $|w|$ of fingerprint w to reach P_{max}. . . . 91
Table 4.4 Required time to extract required minimum-length fingerprints during active times of the day . . . 91

Table 5.1 IoT devices and their connectivity technologies 114

ACRONYMS

6LoWPAN IPv6 over Low-Power Wireless Personal Area Networks

ANN artificial neural network

AP access point

ATM automated teller machine

BLE Bluetooth Low Energy

CERT computer emergency response team

CoI Context of Interest

ConUCON context-aware usage control

CVE Common Vulnerabilities and Exposures

DDoS distributed denial of service

DNS Domain Name System

ECC error-correcting code

EKE Encrypted Key Exchange

FAR false accept rate

FM frequency modulation

FNR false negative rate

FPR false positive rate

FRR false reject rate

GPS Global Positioning System

GRBAC Generalized Role-Based Access Control

HTTPS Hypertext Transfer Protocol Secure

IAT inter-arrival time

IBC Identity-Based Cryptography

ICMP Internet Control Message Protocol

IP Internet Protocol

IoT Internet of Things

IoTSS IoT Security Service

IoTSSP IoT Security Service Provider

JSON JavaScript Object Notation

KDC key distribution centre

kNN k-Nearest-Neighbours

LBS Location-Based Service

LHL Leftover Hash Lemma

MAC Medium Access Control

NAT Network Address Translation

NFC Near-Field Communication

NIC network interface card

NIST National Institute of Standards and Technology

NVD National Vulnerability Database

OSN On-line Social Network

OS Operating System

OVS Open vSwitch

PAKE password-based key exchange

PC personal computer

PDoS permanent denial of service

PIN Personal Identification Number

PoP Proof-of-Presence

PTK Pairwise Transient Key

QR Quick Response

RBAC Role-Based Access Control

RF Random Forest

ROC Receiver Operating Characteristic

RS Reed-Solomon

RSSI Received Signal Strength Indication

RF Radio Frequency

SDN software-defined networking

SGW Security Gateway

SOHO Small Office, Home Office

SSID Service Set Identifier

TCP Transmission Control Protocol

TLS Transport Layer Security

TPR true positive rate

TSF Time Synchronization Function

UI user interface

USOM User Space Object Manager

WPA2 WiFi Protected Access 2

WSN wireless sensor network

ZIA zero-interaction authentication

1 INTRODUCTION

During the past decade the way in which people connect to the Internet has profoundly changed as smartphones have taken over the role as the most prevalent platform for connecting to and consuming Internet content [136]. Whereas in the first wave of the broader adoption of Internet connectivity desktop computers with fixed Internet connections played the main role, the advent of smartphones led to a significant increase in the number of Internet-connected nodes. Subsequently, a major boost in the increase of network connectivity is driven by the so-called Internet of Things (IoT). The term IoT denotes a recent trend in which devices and appliances that traditionally have not been connected to the Internet are increasingly being equipped with smart functionalities and wireless and/or wired connectivity, enabling these devices to be connected to IP networks, allowing them to be remotely monitored and controlled. It is forecast that the adoption of IoT in the forthcoming years will boost the amount of Internet-connected devices, reaching an estimated total of 20.4 billion connected IoT devices in the year 2020 [43].

In the era of desktop computers there were only a few Internet-connected devices with a fairly limited set of application programmes in a typical household, making management of relevant security and privacy settings of the systems feasible (although not easy to perform for regular non-technical persons). Due to this limited connectivity, also the privacy exposure of users and their home environment to external on-line threats was much more limited than in a contemporary IoT smart home, where potentially dozens of Internet-connected devices actively sensing their environment are connected to the local network of the smart home.

The adoption of IoT has consequently led to a drastic increase in the exposure of users' potentially sensitive private information to various security and privacy threats. Modern households might be equipped with dozens of devices with network connectivity that have a rich set of context sensors to sense their surroundings in order to react to the user's actions or adapt their behaviour to environmental conditions. In addition, users typically carry with them one or more smart devices with tens of different applications and apps, each of them possibly having separate sets of configurable settings and security policies. The profound changes that the emergence of smartphones and IoT have had on computing platforms have brought forward new challenges with regard how these systems can be made easily usable by users while at the same time making sure they do not compromise

the security of the system or the privacy of user data being processed by them.

1.1 CHALLENGES

1.1.1 *Context-Aware Access Control*

To protect the security and privacy of users, effective access control mechanisms have to be employed in order to limit privacy exposure according to the preferences of each user. However, as mobile and IoT systems intensively interact with their surroundings and provide context-dependent functionalities, also the access control mechanisms need to take the context into account in their decision-making process. A number of works have therefore addressed the problem of context-aware access control [26, 27] and approaches for allowing users to specify context-dependent access control policies [56, 118, 68]. Approaches for implementing context-aware access control on mobile operating systems have also been proposed [5, 24, 117].

While fine-grained access control policies can be used to realize such controls, their use raises practical usability problems, as the number of policies to be specified, managed, and updated easily grows to be very large. The required effort for doing this easily becomes unreasonably high so that users are not willing to invest the time required for setting up and managing their personal policy sets. In addition, regular users seldom are capable of fully understanding complex security policies and their full privacy ramifications, thereby raising the risk that users make errors in configuring their policies, potentially leading to unwanted privacy compromise. To encounter this challenge a number of works [115, 53, 47] have therefore started to seek ways in which instead of defining a multitude of very fine-grained security and privacy policies, contextual information is used to *learn* and automatically adapt, e. g., authentication requirements based on the context history of the user. This work follows a similar approach: we want to profile and learn about the contextual factors pertinent to the particular use case and utilise this information in adapting security policies or make authentication decisions.

1.1.2 *Context-Based Proofs-of-Presence*

The prevalence of novel contextual sensors in widely-used smartphone models has enabled the introduction of entirely new classes of applications that utilise context information in various ways. Among the most used contextual factors is undoubtedly the *location* of the user, widely available through the use of Global Positioning System (GPS) sensors, or, network triangulation. This has led to the emergence of numerous Location-Based Services (LBSs). However, as all

widely-used positioning technologies rely on the mobile client performing the positioning operation, it is not possible for other entities to verify the correctness of location claims made by clients, opening up the possibility for a wide range of *location cheating attacks*. To encounter such attacks a number of proximity verification approaches have been proposed utilising, e. g., WiFi Received Signal Strength Indication (RSSI) measurements [143], or observing packet header field values on WiFi networks [100].

Also context-based approaches for proximity verification utilising ambient contextual modalities like sound and luminosity sensed directly from the environment have been studied [140, 131]. However, these works consider only a non-adversarial setting related to *relay attack* prevention in zero-interaction authentication (ZIA) use cases, in which both parties, the *prover* making a location claim and the *verifier* verifying it are trusted. This leaves the question open whether contextual information can be used for reliable proofs-of-presence also in settings in which the prover is not trusted.

1.1.3 *Context-Based Authentication*

As discussed above, the increasing number of devices in IoT environments like smart homes containing dozens of installed IoT devices poses challenges with regard to how the security of individual devices is managed. One particular problem is how to manage the authentication of individual devices while providing effective separation between the trust domains of individual users. Conventional approaches like, e. g., PIN code-based Bluetooth authentication relying on manual authentication between devices faces usability challenges and is error-prone, as users need to separately perform authentication with each of the (potentially numerous) IoT devices. Key pre-sharing-based approaches [35, 23, 76, 139] mainly developed for wireless sensor networks (WSNs) on the other hand fail to provide effective separation between the trust domains of adjacent users' smart homes.

Other approaches seek to utilise the proximity of devices to each other to enable co-location based authentication by using context-based information sensed from the ambient environment of the devices as an authenticator. These schemes are based on either sensing WiFi RSSI values [143] or fluctuations in the Radio Frequency (RF)-environment [82]. They have, however, the drawback that their security properties are lacking, or, they are limited to relatively short distances between to-be-paired devices, making their use in real-world IoT settings impractical.

Other contextual modalities like ambient audio have more desirable properties for realising practical context-based authentication, as it allows authentication over longer distances. To this end, an authentication scheme has been presented [124] that is based on using error-

correcting codes to enable audio fingerprinting-based key agreement between two peers located in the same audio environment. However, this scheme fails to address all relevant factors affecting the security of the scheme that are needed to quantify the strength of the resulting authentication secret.

1.1.4 *Security Management of IoT Devices*

In addition to the security challenges mentioned above, the difficulty of effective security management in IoT environments is exacerbated by the fact that the IoT device market has become very fragmented, as hundreds of new device manufacturers are bringing new devices to the market. Many of these manufacturers have little experience in building secure network-connected systems and utilize therefore flawed security designs in their products. On the other hand, many manufacturers are also driven by the desire to bring their products to the market quickly, not leaving enough time to do proper security testing of their product implementations. Both aforementioned reasons lead to a situation in which it is not only possible, but likely that devices with security vulnerabilities are present in users' smart homes. Identifying such devices before they are compromised by potential attackers and taking appropriate countermeasures to contain the security threat that such devices represent is therefore of high importance.

1.2 THE GOAL OF THIS DISSERTATION

The goal of this work is to demonstrate how appropriate context and communications profiling approaches can be used to address the challenges enumerated above in order to provide building blocks for realising more autonomous security decision making in end-user systems. In particular, we target personal mobile devices and IoT devices in smart home settings. Our aim is to make security and privacy management of the increasingly complex and diverse computing environment feasible for regular end-users in the future. In this dissertation we demonstrate how by appropriately profiling information obtained through contextual sensors of devices and their communication behaviour, many security decisions can be automated in a way that makes it easier for regular users to handle the vast complexity of contemporary systems comprising numerous IoT devices and apps. The concrete solutions proposed in this dissertation are related to the automated and context-based adaptation of security and privacy policies of personal mobile devices (Chapter 2), context-based protocols for proximity verification (Chapter 3) and co-presence authentication (Chapter 4), as well as the use of communications profiling for identification and management of IoT devices (Chapter 5).

1.3 OVERVIEW OF CONTRIBUTIONS

1.3.1 *Context-Aware Access Control Framework*

We present the design of ConXsense, a novel context-aware framework for dynamic adjustment of security- and privacy-relevant enforcement decisions on mobile systems. The framework is based on profiling contexts and persons with which the mobile device user interacts with regard to their familiarity. It derives from this profile information features characterising the context which can be used by a machine learning-based classification model to provide predictions about security-relevant properties like the privacy sensitivity of the context and its associated risk of device misuse. Our empirical evaluation based on real-world contextual data demonstrates the effectiveness of the proposed approach in protecting users against threats arising from device theft or misuse, and, privacy leakage by so-called *sensory malware*.

1.3.2 *Resilient Context-Based Proofs-of-Presence*

We empirically analyse a scheme for context-based Proofs-of-Presence (PoPs) that is derived from approaches used in earlier approaches [140, 131] based on real-world context data and demonstrate the feasibility of so-called *context-guessing attacks*. In such attacks a malicious prover uses profiled information about a target context in which the verifier of a PoP scheme is located to *forge* PoPs that the verifier will erroneously accept as genuine Proofs-of-Presence. We then develop an approach utilising context profiling to estimate the *surprisal* associated with individual PoPs and using this information to filter out PoPs that are at risk of being too easy to guess by an adversarial prover. Based on our evaluation data we demonstrate the effectiveness of this *surprisal filtering* approach in protecting against context-guessing attacks. Finally, we introduce an extension to the PoP approach utilising longitudinal context measurements based on luminosity and audio and demonstrate how such measurements can be used to make PoPs more resilient to context guessing in most cases.

1.3.3 *Context-Based Authentication*

Based on a context-based key exchange approach presented in earlier work [124], we develop a *context-based key evolution approach* that aims at gradually increasing the confidence of pairing principals in the authenticity of their counterpart. The approach relies on the fact that only devices that are persistently present in the same context will be able to successfully complete a number of authentication iterations which is required to establish a sufficiently strong authentica-

tion between the pairing parties. We also present a rigorous analysis of the factors that affect the security of context-based authentication. In particular, we quantify the effects that entropy losses incurred by the use of error-correcting codes and the inherent entropy rate present in the context have on the security of the scheme and evaluate it empirically based on contextual measurement data from real-world IoT settings.

1.3.4 *Device Profiling for IoT Security Management*

We present the design of IoTSentinel, a framework that aims at providing a *brownfield* security management solution for networks containing IoT devices with security vulnerabilities. IoTSentinel is based on profiling the communications of *IoT* devices that are newly added to the network and using this for identifying their *device type*. This is required for identifying devices with known security vulnerabilities in order to enforce appropriate traffic filtering measures with which 1) compromise of vulnerable devices can be mitigated, and, 2) in case vulnerable devices are compromised, the rest of the network can be protected against potential attacks utilising the compromised device. We evaluate the effectiveness of IoTSentinel based on empirical data collected from a large set of real-world IoT devices and demonstrate that it can be successfully used for managing threats arising from security vulnerabilities present in *IoT* devices. We also present the design of a prototype system demonstrating the core functionalities of IoTSentinel.

1.4 RELATED PUBLICATIONS

This dissertation is based on several previously published papers as listed below. A full list of all published works of the author of this dissertation is given in chapter 7.

Chapter 2

Markus Miettinen, Stephan Heuser, Wiebke Kronz, Ahmad-Reza Sadeghi and N. Asokan. "ConXsense – Context Profiling and Classification for Context-Aware Access Control". In: *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2014)*. ACM. Kyoto, Japan, June 2014. DOI: 10.1145/2590296.2590337

Chapter 3

Markus Miettinen, N. Asokan, Farinaz Koushanfar, Thien Duc Nguyen, Jon Rios, Ahmad-Reza Sadeghi, Majid Sobhani and Sudha Yellapantula. "I know where you are: Proofs of Presence resilient to malicious provers". In: *10th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2015)*. Apr. 2015. DOI: 10.1145/2714576.2714634

Chapter 4

Markus Miettinen, N. Asokan, Thien Duc Nguyen, Ahmad-Reza Sadeghi and Majid Sobhani. "Context-Based Zero-Interaction Pairing and Key Evolution for Advanced Personal Devices". In: *Proc. ACM Conference on Computer and Communications Security*. Scottsdale, AZ, USA: ACM, Nov. 2014. DOI: 10.1145/2660267.2660334

Markus Miettinen, Thien Duc Nguyen, N. Asokan and Ahmad-Reza Sadeghi. "Revisiting Context-Based Pairing in IoT". in: *Proceedings of the 55th Design Automation Conference (DAC)*. ACM, June 2018. DOI: 10.1145/3195970.3196106

Chapter 5

Markus Miettinen, Samuel Marchal, Ibbad Hafeez, N. Asokan, Ahmad-Reza Sadeghi and Sasu Tarkoma. "IoT Sentinel: Automated Device-Type Identification for Security Enforcement in IoT". in: *Proc. 37th IEEE International Conference on Distributed Computing Systems (ICDCS 2017)*. June 2017. DOI: 10.1109/ICDCS.2017.283

Markus Miettinen, Samuel Marchal, Ibbad Hafeez, Tommaso Frassetto, N. Asokan, Ahmad-Reza Sadeghi and Sasu Tarkoma. "IoT Sentinel Demo: Automated Device-Type Identification for Security Enforcement in IoT". in: *Proc. 37th IEEE International Conference on Distributed Computing Systems (ICDCS 2017)*. Atlanta, GA, USA: IEEE, June 2017. DOI: 10.1109/ICDCS.2017.284

1.5 CONTRIBUTIONS OF THE AUTHOR

The contributions of this dissertation were produced in collaborative research projects where the author had a central leading role. The individual contributions can be detailed as follows.

1.5.1 *Context-Aware Access Control Framework*

The author led the project and designed the context features used for the context classification models and co-designed the overall framework with other project participants. He participated in the requirements gathering and design of the user survey, which was realised in practice and evaluated by co-author Wiebke Kronz. The author spe-

cified the requirements for the data collection software used in collecting evaluation data, organised the data collection campaign and performed the evaluation of the data. The enforcement framework for realising dynamic adaptation of locking time-outs and access to context sensors was contributed by co-author Stephan Heuser. All co-authors participated in writing the paper manuscript [90].

1.5.2 *Resilient Context-Based Proofs-of-Presence*

The author led the project and co-designed the ConXPoP framework. He provided the requirements for the data collection software used for collecting evaluation data, implemented by co-authors Duc Thien Nguyen, Majid Sobhani and Jon Rios and planned and supervised the data collection campaign for acquiring evaluation data in collaboration with co-authors. The author formalised the concept of context-guessing attacks and devised the use of surprisal filtering for mitigating such attacks. He also co-designed the longitudinal context features used for hardening proofs of presence and co-implemented the data analysis and evaluation of the framework together with Duc Thien Nguyen and Majid Sobhani. The author was responsible for authoring the major part of the paper manuscript [88].

1.5.3 *Context-Based Authentication*

The author led the projects related to the underlying papers [89, 94] and co-designed the context-based key evolution framework introduced in the first paper related to this topic [89]. The author co-designed the context fingerprinting approach used in [89] together with Thien Duc Nguyen and Majid Sobhani and contributed requirements for the data collection software, which was realised by Thien Duc Nguyen and Majid Sobhani. The author organised and realised the analysis and evaluation of the collected contextual data together with the aforementioned co-authors. The author was responsible for writing a major part of the first paper's [89] manuscript. In the second paper, the author co-designed the context-based authentication approach and co-developed the formalisation of the relevant factors affecting the security of the context-based pairing approach. The improved fingerprinting approach based on peak detection was developed by Thien Duc Nguyen incorporating some initial contributions from the author. The data collection and evaluation was realised by Thien Duc Nguyen in collaboration with the author. The author was responsible for authoring a major part of the paper [94].

1.5.4 *Device Profiling for IoT Security Management*

The author co-led the project and was responsible for setting up the data collection infrastructure and planning and conducting data collection experiments on IoT devices. The author co-designed the overall architecture with other co-authors. Samuel Marchal designed and implemented the machine learning components for device identification. Ibbad Hafeez implemented the SDN-based isolation solution. All co-authors contributed to the writing of the manuscript [92]. In addition, Tommaso Frassetto supported the implementation of the demonstrator solution [93] for IoTSentinel.

2

CONTEXT PROFILING FOR AUTOMATING SECURITY AND PRIVACY POLICY DECISIONS

The emergence of appified smartphone platforms like Google's Android, Apple's iOS and Microsoft's Windows Phone has led to a rapid growth in the number and complexity of different security and privacy settings that are required for controlling smartphones and applications installed therein. Not only have the smartphones themselves numerous settings that need to be configured, but users also need to consider numerous settings related to the permissions and data sharing by apps that they have installed on their smartphone. Predominant solutions for specifying and maintaining security and privacy settings that reflect the desires of the user are based on manually pre-defined policies. Due to their increasing number and complexity, it is, however, becoming increasingly difficult for users to maintain their policy sets in a meaningful way. The goal of this work is therefore to develop a novel framework for automating security and privacy policy decision making by utilising contextual information that mobile devices can sense with their sensors from the context in order to allow the system to automatically learn and adjust policies based on the contextual situation in which the user is located.

2.1 PROBLEM DESCRIPTION

Configuring and maintaining such large sets of security policies is laborious and requires a considerable amount of expertise and understanding to perform correctly. In many cases, users are not willing to spend substantial amounts of time required to go through and adjust security settings to match their personal preferences. This can result in policies being inadequately configured so that the security and privacy settings of systems and apps do not reflect the true preferences of users. Also, in many cases regular users may not fully understand the detailed consequences of particular policy configurations for their privacy, resulting in erroneous settings that do not reflect the actual preferences of the user with regard to their privacy. A study by Sadeh *et al.* [118] concerning location sharing policies revealed that users often have great difficulties in defining accurate access control policies. In this study, they investigated the ability of users to define policies for controlling the disclosure of their location to other users and user groups like friends, colleagues, etc., and found that the accuracy of location disclosure rules defined by users was initially only 59%. The accuracy increased to 65% after the users were given the chance to

modify their rules after reviewing them based on concrete examples of enforcement decisions resulting from applying the initially-defined rules. In another study by Bauer *et al.* [9], in which the user understanding of data sharing consent dialogues of popular single-sign on services like Google, Facebook and Google+ were studied, the authors found that the *majority* of users did not fully understand which of their data items would actually be shared with a third-party website by the single-sign-on service when the user used it to log in to the third-party-website. This was in spite of the fact that the users were shown an explicit consent dialogue informing them about which data items would be shared by the service. Both above results show that defining, maintaining and understanding the meaning of access control policies is a challenging and likely too laborious and difficult task to be successfully handled by regular users.

What makes the situation even more challenging is that some policy enforcement decisions should be dependent on the context, i. e., the situation and environment in which the user is located at a particular point in time. Capturing all relevant contexts with pre-defined access control rules increases the complexity of the policy set and thus the work required to set up and maintain it even more. It is also questionable, whether it is even possible to enumerate and define all relevant contextual settings in which access control decisions need to be made beforehand.

2.1.1 *Challenges with Straightforward Approaches*

One way to ease the burden of users is to design systems so that they use sets of *default policies* designed by security experts. User involvement would then only be limited to making a selection of what set of default policies to apply. While being an easy choice to users, this approach, however, also has its problems. For one, default policies can never capture the fine-grained personal preferences that users may have, as they are by necessity generalisations and therefore not able to capture all concrete contexts and situations of a user's life. Thus they are limited to using 'reasonable defaults', thereby potentially resulting in settings that do not reflect the true desires of users. On the other hand, understanding the full implications of particular default security policies may not be easy for regular users. Security settings and policies being highly abstract concepts, it is possible that users do not fully understand the consequences that their selected default policies may have, e. g., on the privacy of certain user data. This may result in situations where the actually enforced settings do not correspond to what the users think they do, thereby putting user privacy at risk. This is particularly problematic if policy settings are not as strict the user thinks they are, potentially leading to unwanted disclosure of inappropriate or embarrassing information about the user.

2.1.2 *Goal Setting*

Given the above challenges that contemporary approaches for access control on mobile appified platforms are facing, it is clear that better methods are required for managing the security and privacy settings of users. In this work, therefore, we propose ConXsense, a system that uses profiling of longitudinal observations of the user's behaviour and context together with limited user feedback to automatically *learn* appropriate policy settings for the user to be used in particular situations. This allows ConXsense to realise *usable, user-friendly* but also *accurate* access control. In particular, the envisaged system shall be able to:

- capture relevant features of the users surroundings, i. e., the *context* that influence the users' perceptions on desired access control enforcement decisions,
- autonomously learn relevant places and environments in the user's life, so-called *Contexts of Interest (CoIs)* in order to associate policy decisions with them,
- learn, using minimal user feedback, the desired access control policy decisions to be applied in a particular context.

2.1.3 *Definition of the Term Context*

In this work, we define the term *context* to mean the combination of any ambient physical properties of the surrounding environment that a device can sense with its on-board sensors at a particular point in time. Context can therefore refer to information about the device's position, surrounding other (wireless) devices, as well as direct physical properties of the environment, like, e. g., illumination, audio observations, temperature, humidity, etc. In our approach we seek to utilize information obtained by profiling a device's context over time to build a machine-learning based context model that allows a device to make automated decisions about access control policy enforcement in particular contexts that the device is in.

2.1.4 *User Perceptions of Context*

In an on-line user survey [90], we investigated more than one hundred active smartphone users' perceptions and concerns related to security and privacy issues arising from their smartphone use. The survey sought to identify which contextual factors play a role in users' perceptions with regard to security and privacy. In this survey, two main concerns could be identified, one related to the threat of *device misuse* and the other related to concerns of *privacy exposure*, i. e., the

fear that private or sensitive information about the context is disclosed to unauthorised parties.

The survey revealed that a number of factors primarily influence users' perceptions about these concerns. For one, the presence of other persons and their familiarity to the user were seen as the main factors influencing the perceived risk of device misuse, whereas the familiarity of the place or location as such was the main factor influencing the perceived privacy exposure in particular situations. These findings motivated our selection of primary use cases for ConXsense (as detailed below in Sect. 2.1.5) and also informed the design of the context model by guiding its construction so that it could model both the familiarity of places users often visited, as well as the presence and familiarity of other persons in specific situations, as explained in detail in Sect. 2.3.

2.1.5 *Use Cases*

Motivated by the results of the survey, we selected two prominent use cases that are related to the prime concerns of mobile device users: the fear that their device is misused by third parties, and, the unauthorised disclosure of potentially sensitive contextual information about the user by unauthorised, potentially malicious applications on the user's mobile device. We selected two concrete use cases focusing on these concerns.

2.1.5.1 *Use Case 1: Protection Against Misuse Using a Context-Aware Device Lock*

Mobile devices like smartphones and tablets are increasingly used for access to sensitive information like financial or banking data, personal messaging, on-line shopping, etc. However, reports have shown that many users do not sufficiently protect their devices using protection mechanisms like device locks [21, 132], even though these are readily available on most smartphones and provide effective protection against many threats related to device misuse. This may be due to the fact that device locks are perceived as relatively inconvenient to use, as they require the user to type in an unlocking code several times a day, and alternative solutions like the use of fingerprint scanners for user authentication have only recently started to become widely available on smartphones also in other than the highest price segments.

To reduce the user burden of using device locking for protecting their device against misuse, we adopt the approach first introduced by Gupta *et al.* [47], in which observed context information is used to dynamically adapt the locking time-out of the device lock. By profiling contextual data we develop a model for estimating the *risk*

of device misuse and use it to decide on how quickly the device should lock itself in particular contexts in case it is not used.

2.1.5.2 *Protection Against Sensory Malware*

The term *sensory malware* denotes a class of malicious applications for smartphones. The goal of these malware apps is to use the on-board sensors of the targeted user's smartphone to harvest potentially sensitive information from the user's context. This allows an adversary controlling the sensory malware to aggregate detailed profiles about the user, her behaviour and the environment she routinely lives in. Sensory malware are typically Trojans, i. e., applications with hidden malicious functionality purporting to be a benign app, so that users can be fooled into installing the application on their smartphone.

Notable examples of sensory malware include applications like *Stealthy Video Capturer* [145], *(Sp)iPhone* [81], *Soundcomber* [122] and *PlaceRaider* [138]. Stealthy Video Capturer utilises the camera on the victim's smartphone to capture video images and secretly send them to the adversary, whereas (Sp)iPhone utilises the smartphone's accelerometer to decode vibrations from a nearby keyboard the victim user is typing on to discover the text entered. Soundcomber on the other hand monitors the microphone of the victim's smartphone and uses audio analysis and speech recognition to recover sensitive information like PIN codes or credit card numbers from the victim user's spoken interactions with, e. g., her bank. PlaceRaider utilises the smartphone's gyroscope sensor and camera to harvest sets of images that the adversary can use in reconstructing a 3D model of the victim's surroundings.

In order to protect smartphone users against the threat posed by sensory malware, we adopt an approach in which we limit the access of applications to the contextual sensors of the mobile device in such contexts that are sensitive from a privacy point of view, e. g., a the user's home or workplace. We seek to do this without impacting the legitimate use of the smartphone unnecessarily and therefore do not limit access to sensors in contexts that are not considered privacy-sensitive, e. g., in public places.

2.1.6 *Adversary Model*

In use case 1 concerning protection against device misuse, the adversary is a person in vicinity of the mobile device of the user having potentially physical access to the user's device. The adversary may be malicious, like a thief, honest-but-curious, e. g., a colleague or sibling of the user, or, 'clueless' like a small child. The protection goal in this use case is to minimize the risk that the adversary has access to applications and data on the mobile device of the user. We do this by dynamically adjusting the locking time-out based on the risk of

device misuse in the context. We seek to do so while striking a balance between maximising protection on one hand, and, minimising user inconvenience caused by having to frequently enter a device unlocking PIN or password even in low-risk contexts, on the other hand.

In use case 2 the adversary is an application that is installed on the user's mobile device. We assume that the application has obtained all necessary permissions for access to contextual sensors of the device during its installation. The application may be a malicious Trojan Horse, i. e., sensory malware, or, merely a benign but overly intrusive application. Our goal is to prevent or limit the access of the adversary to contextual sensors of the device in contexts with high privacy exposure, i. e., contexts containing information that the user would want to protect from the adversary. This information may be either *private*, concerning the user personally, or, *confidential*, i. e., other sensitive information not necessarily directly related to the user herself. Typical examples of contexts with high privacy exposure are the user's home containing private information pertaining to the user, or, her workplace with potentially confidential information in the context.

2.2 SYSTEM DESIGN

The ConXsense system is designed to make context-dependent access control decisions based on contextual parameters that a mobile device can observe with its on-board sensors. It utilises machine learning-based classifiers to make predictions about the risk of device misuse and the privacy sensitivity of particular contexts. The classification models are gradually learnt based on ground truth information that the user provides about her security and privacy preferences in particular contexts.

The structure of the ConXsense system is shown in Fig. 2.1. Access control decisions are based on observations of the context obtained by context sensors. The observations are fed to a Profiler that calculates features describing the context. Profiler aggregates profiles of relevant objects (like significant places of the user or other devices that the user encounters) according to the context model described in detail in Sect. 2.3. Profiler periodically uses incoming observations and the aggregated profiles to generate *context features* representing relevant properties of the context at each point in time. The definition of the context features is provided in Sect. 2.3.3.

Context features are forwarded to the Classifier component, which uses them together with ground truth feedback obtained from the user to train and update classification models. The user feedback can be generated either through explicit input like interactions with the device UI, or, implicitly by monitoring the user's actions. Once the classification models have been trained, Classifier uses them to

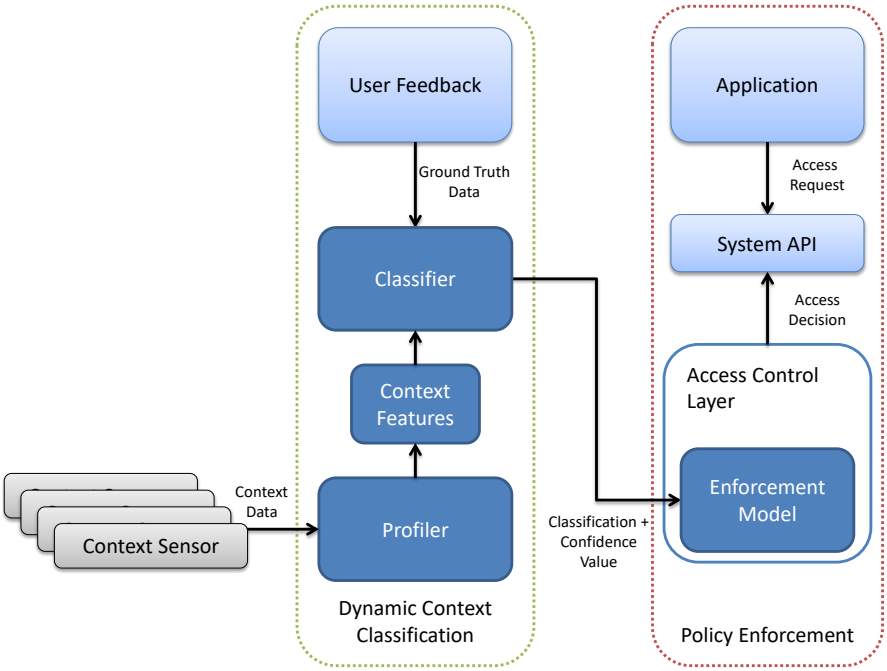


Figure 2.1: The structure of the ConXsense system

provide predictions about a context's security and privacy-relevant properties based on new incoming features from the current context.

The classifier predictions along with their associated confidence values are forwarded to the Access Control Layer, which takes them into account when making decisions about applications' access requests to various functionalities of the System API in accordance with its specific enforcement model.

2.3 CONTEXT MODEL

The ConXsense context model focuses on modelling the surroundings of a mobile device in terms of its location or place in which it is located, as well as the *social context*, i. e., the persons in proximity, and their familiarity to the user. This is because these factors were deemed the most decisive ones influencing the perceptions of users with regard to the risk of device misuse and privacy exposure (cf. Sect. 2.1.4). The model is used for profiling the user's typical contexts as well as deriving context features used in classifying the current context with regard to these properties.

2.3.1 *Profiling Locations and Places*

A core concept of the ConXsense context model are the so-called *Contexts of Interest* (CoIs). They represent places that the user visits often and spends a considerable amount of time in. Typical CoIs for

a user could be, e. g., her home, workplace, grocery store, etc. The ConXsense context model considers two different kinds of CoI: *GPS-based* and *WiFi-based* CoIs. The former correspond to geographical areas determined by their location co-ordinates, and the latter are defined by a characteristic set of WiFi access points usually observed in a specific place. GPS-based CoIs are good in capturing significant places of the user in outdoor areas, whereas WiFi-based CoIs cover also urban indoor areas, where typically GPS reception may not be possible but WiFi access points are available. By using a combination of both CoI types, most significant places that users typically visit can be detected and identified. In the following, we provide a formal definition for both CoI types.

2.3.1.1 *GPS-based CoIs*

Our concept of GPS-based CoIs is based on the notion of *stay points* and *stay regions*, originally introduced by Zheng *et al.* [149] and developed by Montoliu *et al.* [96]. The identification and detection of GPS-based CoIs is based on positioning measurements using the Global Positioning System (GPS). The mobile device measures its position periodically using its GPS sensor. The sequence of measurements is divided into *GPS stay points*, i. e., subsequences of positioning measurements representing the user's visits to distinct places, during which the user stays within a specific distance of r_{sp} from the position of the first GPS measurement in the subsequence. A visit is considered a stay point, if the visit's duration is longer than a minimal duration $t_{min_{sp}}$ and does not contain observation gaps lasting longer than a specified maximum gap length $t_{gap_{sp}}$.

Definition 1 (GPS stay point) *A stay point* $sp = (m_0, m_1, m_2, \ldots, m_n)$ *is a longest subsequence of positioning measurements* $m_i = (lat_i, lon_i)$ *such that*

$$\begin{aligned} \forall i \in \{1, 2, \ldots, n\} : dist(m_0, m_i) &\leq r_{sp}, \\ t(m_i) - t(m_{i-1}) &\leq t_{gap_{sp}}, \\ t(m_n) - t(m_0) &\geq t_{min_{sp}}, \end{aligned} \tag{2.1}$$

where $dist(m_i, m_j)$ *denotes the geographical distance between positioning measurements* m_i *and* m_j, $t(m)$ *the time stamp of positioning measurement* m *and* lat_i *and* lon_i, *respectively, the latitude and longitude readings of a position measurement.*

A stay point's position is defined as the average position of the measurements belonging to the stay point.

Definition 2 (Position of stay point) *The position* $pos_{sp} = (lat_{sp}, lon_{sp})$ *of stay point* $sp = (m_0, m_1, m_2, \ldots, m_n)$ *is defined as the average of all position measurements* $m_i = (lat_i, lon_i)$ *belonging to the stay point, i.e.,*

$$\forall m_i = (lat_i, lon_i) \in sp : \quad lat_{sp} = \frac{\sum_{i=0}^{n} lat_i}{n}; \quad lon_{sp} = \frac{\sum_{i=0}^{n} lon_i}{n} \tag{2.2}$$

The average positions of stay points are aggregated to form *GPS-based Contexts of Interest (CoIs)*. These are rectangular geographical areas of at most $gps_{\max}$ width and length which the user has visited at least $f_{\min_{CoI}}$ times for a total duration of at least $t_{\min_{CoI}}$ minutes.

Definition 3 *A* GPS-based CoI $C = (lat_{\min}, lon_{\min}, lat_{\max}, lon_{\max})$ *is a geographical area delimited by the co-ordinates* $lat_{\min}, lon_{\min}, lat_{\max}$ *and* $lon_{\max}$ *enclosing a set* $S = \{sp_1, sp_2, \ldots, sp_n\}$ *of observed stay points* sp_i *with associated stay point position* $pos_{sp_i} = (lat_i, lon_i)$*, such that*

$$\begin{aligned} \forall sp_i \in S: \quad lat_{\min} \leq lat_i \leq lat_{\max}, \quad lon_{\min} \leq lon_i \leq lon_{\max}, \\ |S| \geq f_{\min_{CoI}}, \\ \sum_{sp_i \in S} dur(sp_i) \geq t_{\min_{CoI}}, \\ lat_{\max} - lat_{\min} \leq gps_{\max}, \quad lon_{\max} - lon_{\min} \leq gps_{\max}, \end{aligned} \tag{2.3}$$

where $dur(sp)$ *denotes the duration of the stay point.*

EXAMPLE. Let us consider a user who regularly commutes between his workplace and home. He also regularly visits a supermarket for shopping and a fitness club. He usually always carries his smartphone with him, which continuously senses the GPS position of the user together with other contextual data.

If the user goes to the supermarket and stays there for 15 minutes, which is longer than $t_{\min_{sp}} = 10\,\text{min}$ and moves there only within a radius of $r_{sp} = 100\,\text{m}$, a stay point sp of duration $dur(sp) = 15\,\text{min}$ is generated. The position of the stay point pos_{sp} will be the average of all position observations pos_i recorded during the stay point visit, most likely located inside or near the supermarket. Waypoints along the user's daily commuting route between his workplace and home will, however, not generate stay points, since the user will not spend a sufficiently long period of time close to the same geographical location.

If the user visits the supermarket ten times and stays there each time for 15 minutes, ten stay points will be generated. These will be aggregated to a GPS-based CoI, since the total stay duration of 150 min is longer than the required $t_{\min_{CoI}} = 30\,\text{min}$ and there are more than the required $f_{\min_{CoI}} = 5$ stay points falling within the area of the CoI. This corresponds to the smallest rectangular area aligned with the GPS co-ordinate system that encloses all of the ten generated stay points, of at most $gps_{\max} = 100\,\text{m}$ width and height.

2.3.1.2 *WiFi-based CoIs*

In order to capture the positions of users better also in urban indoor contexts, where GPS reception often is not available, we extend the notion of stay points and CoIs to indirect positioning information

obtained by monitoring the observable WiFi Service Set Identifiers (SSIDs) at the user's location. This is done by continuously performing WiFi scans that result in snapshots of WiFi SSID observations. Since an SSID can be observed only within the wireless range of its access point, the set of observable SSIDs acts as an indirect indication of the user's location. In the following, we provide the formal definition of WiFi-based CoIs.

Definition 4 (WiFi snapshot) *A WiFi snapshot $S = \{w_0, w_1, \ldots, w_n\}$ is defined as a set of WiFi SSID observations w_i obtained during a single WiFi scan of duration $t_{\max_{wifi}} = 10\,\text{sec}$. $t(s)$ denotes the timestamp associated with the start of the WiFi scan.*

WiFi stay points are sequences of consecutive WiFi snapshots in which each subsequent snapshot's Jaccard distance to the first snapshot is less than or equal to $1/2$. This means that both snapshots must have at least half of the observed SSIDs in common. For a snapshot sequence to be considered a WiFi stay point, the sequence must have a minimal duration of $t_{\min_{sp}}$ and must not have observation gaps longer than $t_{gap_{sp}}$.

Definition 5 (WiFi stay point) *We denote with $sp = (S_0, S_1, \ldots, S_n)$ a WiFi staypoint. It is a sequence of WiFi snapshots S_i such that*

$$\forall S_i \in sp, i = 1, 2, \ldots, n: \quad J_\delta(S_0, S_i) \leq \frac{1}{2} \wedge t(S_i) - t(S_{i-1}) \leq t_{gap_{\max}}, \\ \text{and } t(S_n) - t(S_0) > t_{\min_{sp}}, \tag{2.4}$$

where J_δ denotes the Jaccard distance, which is a measure for the dissimilarity of sets. It is defined for two sets A and B as

$$J_\delta = \frac{|A \cup B| - |A \cap B|}{|A \cup B|} \tag{2.5}$$

The use of the Jaccard distance to distinguish stay points was selected, because it is not uncommon that even SSIDs with good signal strength are occasionally missed by WiFi scans [32]. The use of Jaccard distance allows to compensate for this phenomenon.

We denote as the *characteristic set* of a staypoint a subset of SSIDs that occur in at least half of the WiFi snapshots belonging to the stay point.

Definition 6 (Characteristic set) *The* characteristic *set of a WiFi stay point $sp = (S_0, S_1, \ldots, S_n)$ is denoted with $S_{char} = \{w_0, w_1, \ldots, w_k\}$ and it is the set of all SSIDs w_i that occur in at least half of the WiFi snapshots S_j belonging to the stay point, i. e.,*

$$S_{char} = \left\{ w_i \in \bigcup_{j=0,1,\ldots,n} S_j \in sp \,\middle|\, \frac{|\{S_j | w_i \in S_j\}|}{n} \geq \frac{1}{2} \right\} \tag{2.6}$$

Fixed WiFi access points are likely to be observed at a location each time the user visits it. Reoccurring SSIDs can therefore be used to identify locations and are used to define CoIs. A set of WiFi SSIDs is considered a *WiFi-based CoI*, if the set is a subset of at least $f_{\min_{CoI}}$ WiFi stay points' characteristic sets, and the total duration of these stay points is at least $t_{\min_{CoI}}$.

Definition 7 *A* WiFi-based CoI $C = \{w_0, w_1, \ldots, w_n\}$ *is a set of SSIDs* w_i*, such that*

$$\begin{aligned} \left|\left\{sp_i \in SP \,\middle|\, C \subseteq S_{char_i}\right\}\right| &\geq f_{\min_{CoI}}, \\ \sum_{sp_i \in SP,\, s.t.\, C \subseteq S_{char_i}} dur(sp_i) &\geq t_{\min_{CoI}}, \end{aligned} \tag{2.7}$$

where SP *denotes the set of all stay points and* $dur(sp)$ *the duration of the staypoint* sp*.*

EXAMPLE. Consider the user from the previous example. When he arrives at his workplace, a series of WiFi snapshots $w_0, w_1, w_2, \ldots$ is recorded. Subsequent snapshots that have a Jaccard distance less than 0.5 to the first snapshot form a WiFi stay point sp given that the time difference of the first and last snapshot is at least $t_{\min_{sp}}$ seconds and there are no observation gaps longer than $t_{gap_{\max}}$. The characteristic set of SSIDs S_{char} of this stay point contains the SSIDs of typical access points observed in the workplace. During subsequent visits of the user to the workplace, more stay points with the same characteristic set will be generated. This set is considered a WiFi-based CoI, if at least $f_{\min_{CoI}}$ such stay points are observed and the total visit duration of these stay points is at least $t_{\min_{CoI}}$.

Given the above contexts, we define the *location context* of the user to consist of all the CoIs that the user is visiting at the time. We define visits as follows.

Definition 8 (Visits to GPS-based CoIs) *To capture visits of users in particular frequently-visited places in terms of their geographical position, we define a visit* $V_C = (pos_0, pos_1, \ldots, pos_n)$ *to a GPS-based CoI* $C = (lat_{\min}, lon_{\min}, lat_{\max}, lon_{\max})$ *as a sequence of consecutive position observations* pos_i *falling within the area of* C*, having a maximum time difference between observations of at most* $t_{gap_{\max}}$

Definition 9 (Visits to WiFi-based CoIs) *To capture visits of users in particular frequently-visited places in terms of their observed RF environment, we define a visit* $V_C = (S_0, S_1, \ldots, S_n)$ *to a WiFi-based CoI* $C = (w_0, w_1, \ldots, w_k)$ *to be a sequence of consecutive WiFi snapshots* S_i *having a Jaccard distance less than* $\frac{1}{2}$ *to the CoI, i.e.,* $\forall S_i \in V_C : J_\delta(C, S_i) \leq 0.5$*. The time distance between consecutive observations being at most* $t_{gap_{\max}}$*.*

We denote the set of all visits V_C to context C with $\mathcal{V}_C$.

Definition 10 (Location context) *The location context* L_t *of a user at time point* t *is the set of all CoIs the user is visiting at time* t*.*

2.3.2 *Profiling the Social Context*

For capturing the presence of people in the context, ConXsense considers Bluetooth devices like smartphones which are typically carried by people and can be sensed remotely up to a range of ca. 30 metres by scanning the Bluetooth RF environment. Bluetooth has also earlier been successfully used in ubiquitous computing literature to model the presence of people in the context (cf., e. g., [99]). We focus on mobile devices that people typically carry with them, e. g., smartphones, headsets and PDAs by filtering Bluetooth devices by their type, discarding observations of clearly stationary devices like printers.

Bluetooth may not always be able to reliably sense the presence of all persons in the context, as not all of them are likely to keep the Bluetooth radios of their devices enabled, or simply don't carry it with them. However, ConXsense uses the observations in a probabilistic fashion to make predictions about the type of context the user is in, not to trigger specific actions based on the presence or non-presence of particular devices in the context. Especially in public places where many people are present, the likelihood of being able to observe Bluetooth devices is high and thereby a good factor in classifying the context.

ConXsense models the social context in terms of Bluetooth devices that are detected in the user's context. Social connections are further profiled using the notion of *encounters*, i. e., occasions during which the user's device has been in proximity of other Bluetooth devices.

Definition 11 (Encounters) *An* encounter $E_d = (b_0, b_1, b_2, \ldots, b_n)$ *with a device d is a sequence of Bluetooth observations b_i of device d in which consecutive observations have a time difference of at most $t_{gap_{enc}}$. $\mathcal{E}_d$ denotes the set of all encounters E_d with device d.*

As discussed in Sect. 2.1.4, the familiarity of persons present plays a role in the user's perception of the risk of device misuse in a context. We therefore also model the familiarity of observed devices as follows.

Definition 12 (Familiar devices) *A device d is defined to be* familiar, *if the device has been encountered at least $f_{\min_{enc}}$ times and the total duration of the encounters is at least $t_{\min_{enc}}$, that is, device d is familiar, if*

$$\begin{aligned} |\mathcal{E}_d| &\geq f_{min_{enc}}, \text{ and} \\ \sum_{E_d \in \mathcal{E}_d} dur(E_d) &\geq t_{min_{enc}}, \end{aligned} \tag{2.8}$$

where $dur(E_d)$ denotes the duration of encounter E_d.

Definition 13 (Device context) *The* device context D_t *at time point t is defined as the set of devices d that are encountered during time point t, i. e.,*

$$D_t = \{d | \exists E_d \in \mathcal{E}_d \, (E_d = (b_0, b_1, \ldots, b_n) \wedge t(b_0) \leq t \leq t(b_n))\} \tag{2.9}$$

EXAMPLE. For our example user, familiar devices typically would be the mobile phones of his family members or colleagues, as it is likely that these devices will be encountered more often than $f_{\min_{enc}} = 5$ times and the total duration of these encounters is likely to exceed $t_{\min_{enc}} = 30\,\text{min}$.

2.3.3 *Context Features*

Based on the context model we define following features shown in Tab. 2.1 to be used by the machine learning model for context classification. The features are calculated by Profiler and labelled based on ground truth provided by user feedback. The feature values are forwarded to Classifier, which uses them to train a machine learning-based classifier for predicting security-relevant properties of the context as described in Sect. 2.4.3.

Features f_1 to f_4 model the location context of the user, measuring both the total visit time as well as number of visits the user has paid to the CoIs of the current location context (f_1 and f_2 for GPS-based and f_3 and f_4 for WiFi-based CoIs), thus providing a measure for the familiarity of the place the user is currently located in. Features f_5 to f_8 on the other hand aim at providing a characterisation of the social context of the user. They measure the number of Bluetooth devices observed in the context and how many out of those are *familiar* according to Def. 12. Features f_7 and f_8 measure the average encounter time as well as the average number of encounters the the user has had with familiar devices in the current context. This aims at modelling how familiar the persons in the context are to the user.

2.4 EVALUATION

2.4.1 *Data Collection*

To evaluate the ConXsense framework on real empirical data capturing the contexts and perceptions of real smartphone users, we implemented a Context Collector app for Android that was was installed on the smartphones of test users. The Context Collector app recorded the position and context data once every 60 seconds, which was a reasonable trade-off between sufficient granularity of context information and battery lifetime of the used smartphones. Our target was to achieve a battery lifetime of at least 12 hours (a full working day). The collected context data included the GPS position of the device, nearby Bluetooth devices and WiFi access points, acceleration sensor readings, as well as information about user actions and her interactions with apps on the smartphone.

The Context Collector app also provided a GUI shown in Fig. 2.2 for collecting ground truth information about the user's perceived

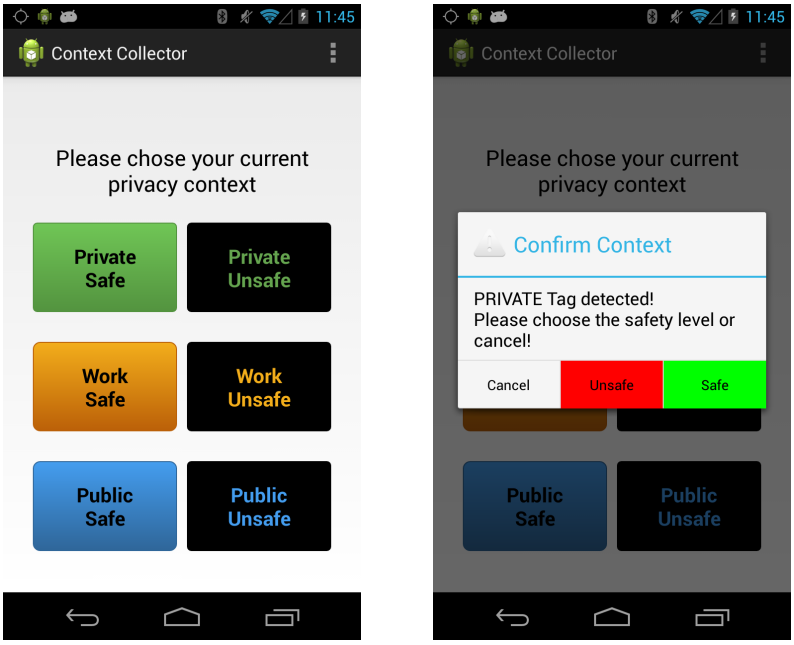
Table 2.1: Context features derived based on the context model

FEATURE	DESCRIPTION
f_1	max total visit time of any GPS-based CoI in L_t
f_2	number of visits to GPS-based CoI in L_t with max total visit time
f_3	max total visit time of any WiFi-based CoI in L_t
f_4	number of visits to WiFi-based CoI in L_t with max total visit time
f_5	number of Bluetooth devices in D_t
f_6	number of familiar Bluetooth devices in D_t
f_7	average total encounter time of familiar devices in D_t
f_8	average number of total encounters with familiar devices in D_t

L_t: Location context (Def. 10) at time t

D_t: Device context (Def. 13) at time t

risk of device misuse and privacy exposure. The users were asked to use the Context Collector GUI to regularly label the current context they were in as "safe", i. e., having a low risk of device misuse, or, "unsafe", i. e., having a high risk of device misuse. To capture the aspect of privacy exposure, users were also asked to label the current context with one of the labels "home", "work" or "public" to capture whether the privacy exposure of the context was due to private (home) or confidential (work) information in the context, or, whether the context was not considered to expose privacy-sensitive information (public). The meanings of these labels and how to apply them were given to the test participants beforehand in order to avoid misunderstandings. The users were encouraged to provide feedback regularly, especially when entering or leaving contexts. To facilitate a more convenient user interaction for providing feedback, we provided users also with Near-Field Communication (NFC)-based context labels labelled 'Private', 'Work' and 'Public' that users could attach to objects in contexts they frequently visited. By touching the NFC tag with the user's NFC-enabled smartphone the GUI of the Context Collector app was automatically brought to the foreground and the corresponding feedback about the privacy sensitivity level recorded. A prompt as shown in Fig. 2.2b additionally requested the user to provide feedback about the level of risk of device misuse for the current context.



(a) GUI for providing explicit user feedback using feedback buttons.

(b) Feedback response dialog after using a 'Private' NFC tag for providing user feedback. The user is asked to provide additional information about the level of risk of device misuse.

Figure 2.2: GUI of the Context Collector app used for collecting ground truth information about the risk of device misuse and privacy sensitivity of the context. 'Safe' indicates a context with low risk of device misuse, whereas 'Unsafe' indicates high risk. 'Private' and 'Work' indicate contexts with high privacy sensitivity, and 'Public' indicates a context not considered sensitive from a privacy point of view.

2.4.2 *Dataset*

We collected context and ground truth data from a set of 15 users coming from technical and non-technical backgrounds. Users provided data over a period of 68 days, on 56 days per user on average, resulting in a dataset containing data from 844 distinct user days. On average users provided ground truth feedback labels for contexts on 46 days, resulting in a ground truth dataset containing 3757 labelled data points. Each user provided at least 50 or more feedback labels. In a deployment setting, this would roughly correspond to 2-3 feedback labels given per day over a period of three weeks, which seems like a manageable burden for users. After this initial training period the need for further user feedback would significantly diminish, and further user interaction could be limited to giving occasional corrective feedback in cases where Classifier would provide incorrect predictions about the security and privacy properties of particular contexts.

2.4.3 *Context Classification*

To implement the Classifier component for context classification we used the Weka data mining toolkit [50], and its k-Nearest-Neighbours (kNN), Naïve Bayes and Random Forest classifier implementations. The kNN classifier is an *instance-based learning* algorithm that bases its prediction on comparing the testing data point to the k nearest labelled data points in the training dataset. The predicted label is the most frequently occurring class label in this set. The Naïve Bayes classifier is a simple probabilistic model that tries to estimate the most likely class label given a set of input features. Naïve Bayes classifiers have been successfully applied, e. g., in e-mail spam detection, where the task is to distinguish spam e-mail from benign mails [119]. The Random Forest classifier [16] is an ensemble learning method based on training a number of different decision trees by randomly sampling the training data and using voting to determine the most common prediction result of the decision trees as the output of the classifier.

For each test participant we trained two binary Classifier instances using the user's labelled context data vectors as input. One was used for for predicting the risk of device misuse in the current context (i. e., is the context "safe" or "unsafe") and the other one for predicting the sensitivity of the context (i. e., is the current context "sensitive" or "public"). As we assume that the mobile device by default applies the most restrictive enforcement mechanisms to protect user data and privacy by using a very short time-out for the device lock and blocking third-party apps' access to environmental sensors of the device, the task of the Classifier component therefore is to identify such contexts and situations in which these restrictive enforcement measures

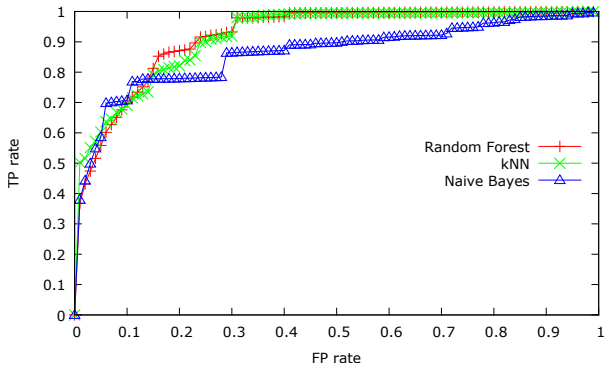


Figure 2.3: Average ROC curves showing performance of classifying contexts with a low risk of device misuse, considering those users who provided at least five ground truth feedbacks per context class.

can be relaxed without increasing the risk of device misuse or privacy compromise too much in order to increase the usability of the device.

Even though it would have been preferable to evaluate the classification results by directly obtaining user feedback on concrete classification decisions in real-time, we had to resort to off-line evaluation, as we wanted to have the opportunity to experiment with different machine-learning algorithms for context classification, and staging the experiment separately for each of the algorithms was not possible due to resource constraints.

2.4.3.1 *Device Misuse Protection*

The average performance of the classifiers in identifying contexts with a low risk of device misuse is shown in Fig. 2.3. It shows the true positive rate (TPR) and false positive rate (FPR) for different confidence values required for a measurement to be classified as "safe" by the classifier. The TPR denotes here fraction of measurements in contexts labelled as "safe" that are correctly classified as "safe", whereas FPR denotes the fraction of measurements in contexts labelled as "unsafe" that are falsely classified as "safe".

All three classifiers perform reasonably well in identifying "safe" contexts. The classifiers achieve a TPR or 70 % at a moderate FPR of 10 %. This would mean that our approach could be used to reduce about 70 % of unnecessary authentication prompts in contexts with low probability of device misuse, whereas only once in ten cases the device locking mechanism would be relaxed while the device is in a context with a high probability of device misuse. This would mean that, e. g., a potential thief would have a success probability of merely 10 % in finding a device in an unlocked state, in a context with high device misuse probability. This is significantly better than the current situation in which many people choose to use no device lock at all, due to the usability penalties imposed by the continuous bur-

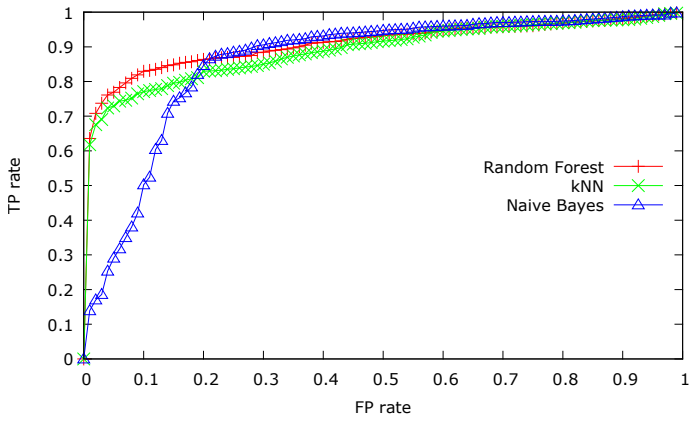


Figure 2.4: ROC curves of classifier performance in identifying public contexts in which sensory malware protection can be relaxed.

den of having to enter a PIN code or password each time the user wants to use the device. The TPR of 70 % also clearly outperforms recent proposals like proposed by Riva *et al.* [115] utilising *progressive authentication*. They report only a reduction of 42 % in the amount of unnecessary authentication prompts presented to the user when applying their approach.

2.4.3.2 *Protection Against Sensory Malware*

In our scenario we assume that access to the ambient context sensors of the mobile device can be granted to third-party applications in contexts that do not contain sensitive information about the user. In our test setting, such contexts were labelled as "public", whereas in contexts labelled as "sensitive" the access should be denied due to the potential danger of sensory malware leaking sensitive information contained in the context to unauthorised external parties. The task of the classifiers in this scenario is therefore to distinguish between these two types of contexts. We therefore measure the TPR as the fraction of measurements labelled as belonging to the "public" class that are also classified as such. Accordingly, FPR in this setting denotes the fraction of measurements performed in "sensitive" contexts (like home or work) that are incorrectly classified as "public". The performance of the classifiers in dependence of the confidence level required to label a context as "private" is shown in Fig. 2.4.

In this use case, there is a clear difference between the performance of the classifiers. The Random Forest and k-Nearest-Neighbours achieve a TPR of approximately 70 % with a very low simultaneous FPR of 2 % to 3.5 %. This means that access to context sensors would be (erroneously) relaxed in less than 3.5 % of the cases when the device is situated in a sensitive context. This would consequently severely limit the ability of a sensory malware to exfiltrate potentially

sensitive information about the user from contexts with high privacy exposure.

The TPR of 70 % implies that in 70 % of the cases when the mobile device is located in a public place with low privacy exposure the system would automatically relax the privacy protections and permit access for third-party apps to sensory information, thereby improving the user experience and utility of such applications. The remaining 30 % of cases in which the privacy protections are not automatically relaxed by the system, this can be handled by offering the user a manual override functionality for temporarily granting access to apps for sensory context information. A significant number of use cases in which contextual information is used by third-party mobile apps involves the active involvement of the user, e. g., when a navigation app is used to find directions to a particular location. Override mechanisms for granting access to sensory data are therefore relatively straightforward to integrate in the application usage flow, as the attention of the user is already focused on the interaction with the particular application. In addition, such override events can effectively be used as additional ground truth feedback when re-training the classification model for improving classification accuracy.

2.5 ENFORCEMENT

Enforcement of access control decisions made by the ConXsense framework is realised by the *FlaskDroid* architecture [18], as described in detail in [90]. FlaskDroid is a fine-grained mandatory access control framework for the Android mobile operating system. FlaskDroid works by instrumenting components that provide access to sensitive resources like the SensorService, which controls access to on-board sensors, as *User Space Object Managers (USOMs)*. USOMs grant or deny requests to the resources they protect based on pre-defined access control rules. However, FlaskDroid supports also *conditional access control rules* with the help of ContextProviders that evaluate the current context at runtime and enable or disable rules according to context-specific conditions.

ConXsense is integrated with FlaskDroid with a ConXsense-specific ContextProvider that uses the classification result and associated confidence values provided by the Classifier component of ConXsense (cf. Fig. 2.1) to activate or deactivate specific access control rules. Such rules can be used, e. g., to either grant or deny third-party applications access to environmental sensors in order to protect against the threat posed by sensory malware.

The activation and deactivation of access control rules can be fine-tuned by specifying rule-specific thresholds for the confidence of context classification provided by Classifier. The thresholds can be selected, e. g., by specifying an upper acceptable bound for the false

positive rate of the system (cf. Figs. 2.3 and 2.4) and using the corresponding classification confidence value achieving this FPR performance as the confidence threshold. It is to be noted that thresholds can be sensor- or resource-specific. This means that for sensors providing particularly sensitive information like, e. g., the GPS sensor that provides information about the detailed position of the user, a higher threshold on the classification confidence could be applied in order to protect the privacy of the user, whereas for sensors like a magnetometer that provides less sensitive information about the user, a lower confidence level would be sufficient.

2.6 RELATED WORK

2.6.1 *Use of Contextual Data for User Profiling*

Context sensing has been use in numerous tasks for profiling users and their relevant life circumstances. For example, Madan *et al.* [78] investigated the use of context measurements sensed with mobile devices for predicting the health status of individuals carrying mobile devices. Factors incorporated in their model included nearby Bluetooth devices, WiFi access points as well as traces about users' communication behaviour. Eagle and Pentland [33] pioneered a study using Bluetooth sensing as a means for establish social proximity for the purpose of allowing users to discover or be introduced to other users sharing common interests or similar properties in their user profiles. In a similar way, ConXsense utilises the use of Bluetooth for proximity sensing of the social situation, with the target of distinguishing situations in which only well-know persons are present from presence in public places in which often a number of unknown users are present.

2.6.2 *Context-Aware Access Control*

Context awareness has been included in a number of access control models. Covington *et al.* [26] introduced *CASA (Context-Aware Security Architecture)* which models security-relevant context or state of the ambient environment with the help of *environment roles*, which is a component of the Generalized Role-Based Access Control (GRBAC) model [98], an extension to the traditional Role-Based Access Control (RBAC) models [120]. Another extension of RBAC by Damiani *et al.* [27], entitled *GEO-RBAC*, uses *spatial roles* to model location as a factor for making access control decision. In contrast to ConXsense, however, all of these approaches require the user to specify the details of used policies including contextual parameters defining the environmental roles or, locations comprising the spatial roles used for enforcement.

Hull *et al.* [56] address the problem of policy specification in their *Houdini* framework, in which they propose to use *policy templates* to ease the burden of policy definition for regular non-expert users. However, even in their system the final decision about concrete policy settings remains with the user. In a similar effort to improve the accuracy of policy sets defined by end-users, Sadeh *et al.* [118] developed an approach in which the process of specifying user policies is facilitated by allowing users to audit concrete enforcement situations based on their currently defined policies and allowing them to refine the policies based on this. However, their results suggest that it is in general difficult for regular users to define accurate policies that reflect the desires of the user beforehand, leading to relative low accuracy in policy enforcement in concrete contextual situations. To address this inherent difficulty of defining accurate policies, Kelley *et al.* [68] introduce in a follow-up work an approach using *user-controllable policy learning* in which the system interacts with the user, providing audited feedback about concrete enforcement decisions taken with the user's policy set and allowing the user to either accept or reject the changes with the goal of incrementally improving the accuracy of the user's policy set. However, this approach requires regular interaction of the user with the system and considerable effort to end up with a policy set that accurately can reflect the user's policy desires.

In contrast to all of the above systems supporting context-based access control, the advantage of ConXsense is that it autonomously learns relevant context definitions used for enforcement without the need for the user to explicitly define her policies. Ground truth for training the system is given by the user in terms of concrete contextual situations the user is situated in, whereby the decisions to be made are intuitive to understand for the user.

2.6.3 *Context-Based Access Control Enforcement in Mobile Systems*

Bai *et al.* [5] present a context-aware usage control (ConUCON) model for the Android Operating System (OS) which is an extension of the UCON usage control model [105, 106] incorporating the notions of spatial and temporal contexts in the usage control model. In another approach, Ongtang *et al.* [103] present *Saint*, a framework extending the system-provided access control model of the Android OS to incorporate run-time control over inter-process interactions. Saint includes also the possibility to use context-based factors for such access control decisions.

Another context-based access control framework for the Android mobile OS is *CRêPE* by Conti *et al.* [24]. It extends Android's standard permission-based usage control model in a way that it allows access control decisions to be conditioned on contextual factors observed by sensors of the mobile device or derived from logical inputs includ-

ing both context sensors and other information about the device's or user's state. Also CRêPE requires the context-based access control policies to be defined by the user beforehand.

The *MOSES* system by Russello *et al.* [117] uses virtualisation to enable the separation of applications and data on the same smartphone through the use of security profiles. To facilitate transitions between different security profiles, *MOSES* utilises criteria on distinct context parameters to trigger the invocation of a particular security profile. Context criteria can be defined as boolean expressions over measurements and readings of physical context sensors of the mobile device or values provided by logical sensors providing higher-level inferred status information about the device's state based on processing of raw sensor information, e. g., detecting that the state of the device user is 'walking' based on an interpretation of the device's accelerometer readings.

While there have been substantial research activities into frameworks allowing contextual factors to be taken into account in access control decisions on mobile device platforms, all of these models require users to specify relatively detailed context policies in advance. As discussed in Sect. 2.1, managing such policy sets is a daunting task for regular users due to the complexity and large number of policies that need to be configured for capturing accurately the true intentions of users. None of the above-mentioned works on context-based policy enforcement on mobile devices have yet sufficiently addressed this problem but have focused more on the technical aspects of access control enforcement. In contrast, this work seeks to find mechanisms with which appropriate profiling approaches and learning from the user's context history can replace the need for explicit fine-grained policy definition by the user.

2.6.4 *Usable Access Control*

Most of the context-based access control frameworks presented above require users to specify fine-grained rules or policies, which is tedious and presents many usability challenges for users are guided to specify their policies. To address this issue a number of frameworks seek to improve the usability of access control systems by using machine-learning approaches to *learn* appropriate settings and make access control decisions based on dynamic evaluation of situations based on this.

Riva *et al.* [115] present an approach called *progressive authentication* in which numerous contextual cues are merged and continuously evaluated in order to determine whether to require re-authentication of the user on her mobile device or not. The approach is based on estimating the likelihood of whether the user is still in proximity of the mobile device and use these estimates to trigger re-authentication

prompts if required. The goal is to minimise the number of unnecessary authentication prompts displayed to the user in order to improve the usability of the system.

The *CASA* system by Hayashi *et al.* [53] uses contextual factors like the location of a mobile device to probabilistically adapt the used active authentication measures in order to provide at all times sufficient confidence in the authenticity of the user. The reasoning is that in some locations less stringent authentication measures are required to obtain sufficient evidence about the authenticity of the device user. While this approach is closely related to the approach taken in this work, it differs from it in the sense that it merely considers the location as such as a factor and does not use its familiarity nor the presence of other devices into account when deciding which active authentication measures to use. However, as our user study in [90] shows, these factors tend to have significant impact on the way in which users perceive the security and sensitivity of contexts and need therefore to be taken into account when evaluating the security policies to be applied in specific contexts.

2.6.5 *Context Identification*

Kang *et al.* [67] introduced an algorithm for identifying places of interest for users. It is based on time-based clustering of location observations by identifying frequently visited locations at which users stay for a minimum amount of time. In their scheme, user location is determined by observing beacon messages from WiFi access points with known co-ordinates and calculating a location estimate by averaging the locations of observed beacons using a centroid tracking scheme. A similar time-based clustering approach to identify significant locations of users was introduced by Zheng *et al.* [149], however, utilising GPS location data. Zheng *et al.* introduced the notion of a *stay point* denoting a geographical region within the user stays for a predetermined minimum amount of time. Montoliu *et al.* [96] extend the notion of stay points to *stay regions*, i. e., geographical areas encompassing a number of different stay points by aggregating stay point observations over time and clustering these to determine places of interest for the user. Our notion of GPS-based contexts of interest is a slightly adapted version of stay regions.

Our WiFi-based contexts of interest are inspired by the work of Dousse *et al.* [32], which introduced the notion of using sets of WiFi access points observed at a particular point in time and their signal strengths to represent distinct places. Our approach is a simplified version of this approach in that it does not consider the actual signal strengths of observed WiFi access points but focuses on the set of characteristic access points (APs) that define the WiFi-based CoIs. This is sufficient for our purposes, as we only need to identify the

presence of the user in a familiar environment for probabilistically adjusting the device's security policies. The exact position of the user is not required to determine presence, which is why using the set of observable WiFi access points is good enough as an indicator of location for our purposes.

2.6.6 *Context-Aware Policy Adaptation*

The work of Gupta *et al.* [47] was the first to use context profiling for the purpose of adjusting security policies. They used the notion of CoIs and device familiarity to make inferences about the 'safety' of particular contexts. Their system was based on a simple heuristic to determine familiarity of devices and places, which would be discounted over time to 'forget' devices and places that the user does not encounter actively. The model applied fixed thresholds to distinguish different types of contexts, which makes it inflexible to accommodating different user behaviours and context types. The present work is inspired by the basic approach of this work, but is based on evolved underlying concepts, both in the way contexts of interest and device familiarity are defined, but also in how the probabilistic reasoning based on contextual measurements is performed. Whereas the earlier work relied on absolute thresholds to be defined beforehand, our approach utilises machine learning-based prediction models. This allows the system to accommodate habits, the environment and security and privacy preferences of individual users in a flexible and user-friendly manner.

2.7 SUMMARY AND CONCLUSIONS

In this chapter, we introduced ConXsense, a framework for context-based adaptation of security and privacy policies on mobile devices. We targeted two threat scenarios that were identified based on a user study to reflect the primary concerns of mobile device users: the threat of *device misuse* by unauthorised third parties, and, the risk of privacy exposure of contextual information due to the threat of sensory malware. We devised an approach that utilises contextual information that a mobile device senses from its environment to profile significant places of the user, so-called Contexts of Interest (CoIs), as well as to model encounters with other users and estimate their familiarity to the user. Based on such contextual factors we developed a machine learning-based model for estimating security-relevant properties of the user's contexts like the *risk of device misuse*, and, its *privacy sensitivity*. By modelling these factors, we seek to mitigate security and privacy risks. First, the risk that an attacker like a thief or other unauthorised parties obtain possession of the user's mobile device while it is in an unprotected state, thereby providing the at-

tacker access to private data of the user and functions of the mobile device. Second, the risk that a new type of malicious software, so-called *sensory malware* obtains and exfiltrates privacy-sensitive data from the surroundings of the user, thereby compromising the privacy of the user.

In developing our framework we seek to strike a balance between security and privacy considerations on one side and usability and user-friendliness of the solution on the other. To protect the user's device from misuse we therefore introduce a dynamic device locking scheme which adjusts the device lock's locking time out based on the estimated risk level of device misuse in a particular context. In high-risk contexts the locking time out is shortened, while in low-risk contexts like the user's home the locking time out can be long, so that the user does not need to enter unlocking credentials when using the device, thereby improving the usability of the device lock. This is important, since many users refrain from using a conventional device locking functionality at all, as it is so cumbersome to use when frequent authentication prompts require the user to repeatedly enter unlocking passwords to the device, leaving the user's device constantly at risk.

Similarly, to protect the user's privacy we can limit the user's privacy exposure by restricting third-party providers' applications' access to contextual sensors in contexts that contain sensitive information from the user's privacy point of view. This effectively mitigates the risk that a sensory malware application can harvest sensitive information from the user's context. On the other hand, we do not want to limit the access to contextual information too much, as many benign applications require it for being useful. Therefore we limit the access to sensory information to contexts with high privacy exposure, while allowing it in contexts with low privacy exposure like public places.

The emergence of mobile devices and IoT is increasing the exposure of users to on-line systems as the number of devices and sensors gathering information about the user and her surroundings is growing. This makes it necessary to devise effective measures for controlling how and which information is shared about the user under which circumstances. As discussed in Sects. 2.6.2 and 2.6.3, a number of approaches has already been proposed for enabling context-aware access control. However, all of these approaches have usability challenges, as it is not realistic to assume that regular users would be able to or willing to use a considerable amount of time to just specify and maintain a comprehensive set of policies or access control rules for controlling the minutiae of how they wish their presence in cyberspace to be represented and what information is to be shared about them in particular circumstances. Consequently, systems should be able to *learn* from the user's behaviour patterns what the security and

privacy preferences of the user are and use this learnt knowledge in enforcing security policies on behalf of the user.

This work represents a first step towards an autonomous approach for security and privacy enforcement. Based on the above use cases we demonstrate how contextual data sensed by the environmental sensors of a device can be used in realising access control that dynamically adapts to particular user-specific situations and accommodates users' perceptions about security-relevant properties of the context. This approach allows the system to strike a balance between enforcement of enhanced security and privacy for the user and the usability of the system. As the learning process of the system can be embedded into the normal usage patterns of the device, it is able to capture the user's desires without the need to go through a tedious process of specifying and refining detailed policies and explicitly specifying the exact circumstances under which particular access control decisions should be made.

This approach towards security policy definition and enforcement can find application also in other use cases requiring access control enforcement. In this work we concentrated on mobile device-related scenarios, but the presented approach could be extended to novel application areas like smart home IoT. Contextual factors play an important role in the assessment and decision making of access control enforcement in many IoT-related scenarios. Also the abundance of sensors present in IoT devices can offer entirely new opportunities for sensing and modelling contextual situations taking the whole ensemble of IoT devices present in, e. g., a smart home into account.

As the proposed framework is making access control decisions in an automated fashion on behalf of the user, it is vitally important for the acceptance of this approach that it communicates its decisions and the reasons behind its reasoning to the user in a way that is understandable. It is also important to provide appropriate mechanisms for the user to override and / or correct inferences that were incorrect or based on imprecise contextual inputs. How to realise this in the best way in practice is still an open challenge. It requires the development and evaluation of appropriate user interaction methods for communicating the state of the system in an unobtrusive but easily accessible way to the user thus enabling her to be in control of her system at any point in time and make directly necessary adjustments if required. Further research in this area around concrete use cases is necessary for addressing this challenge.

3

CONTEXT PROFILING FOR RESILIENT PROOFS OF PRESENCE

Contemporary mobile devices like smartphones are equipped with positioning technologies like the Global Positioning System (GPS) or network triangulation for enabling devices to determine their geographical position. This has led to many new services and smartphone applications that actively utilize positioning information. Online Social Networks (OSNs) like *Facebook*[1] intensively utilise location "check-ins" to enrich their service, and services like *Foursquare*[2] use user location to connect users with local businesses like restaurants and shops. This has prompted a number of business owners to offer concrete benefits like rebates, free vouchers or even cash to the most active registered users of such services visiting their restaurant or shop.

The business model of such LBSs is based on the assumption of trustworthiness of mobile device users. However, as an increasing number of such LBSs are on the rise, also the incentives for users to engage in *location cheating* for obtaining personal benefits are growing. Misbehaving users may try to obtain unjustified advantages by repeatedly fabricating false location check-ins. In fact, "fake location" apps are already available for popular mobile device platforms that can help users to cheat about their true location.

3.1 PROBLEM DESCRIPTION

The inherent problem with most currently deployed positioning technologies is that the determination of location has to be performed by the client device itself; it is very difficult for external entities to verify whether a location claim of a client is in fact genuine or not. Therefore there is a need for *location proofs*, i. e., methods for verifying the correctness of location claims that clients present to an LBS. Similarly, in peer-to-peer settings there may also be a need for location proofs, e. g., in order to allow clients to control their visibility to others. For example, in some settings, a device might be willing to reveal its location only to such peer devices that are co-located with it in the same location [100]. To do this, the peer devices need to present a *proof of co-presence* to prove that they are in the same location.

1 https://www.facebook.com
2 https://www.foursquare.com

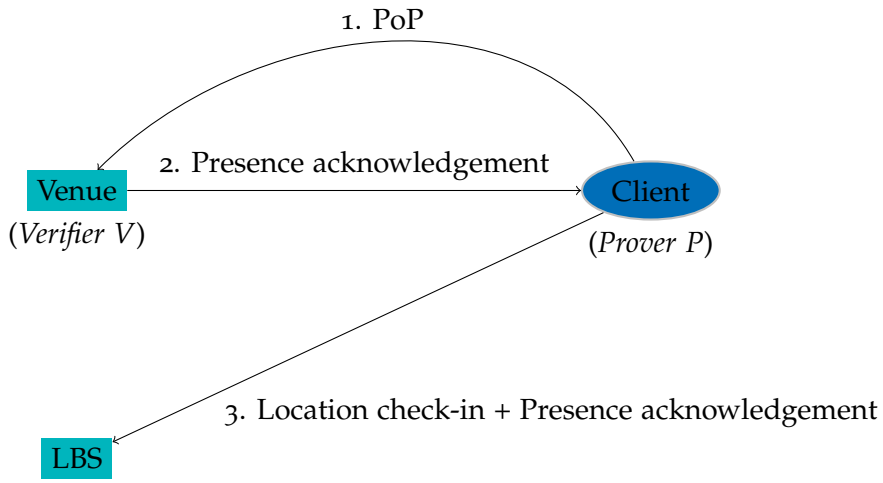


Figure 3.1: Verification of *check-ins* in location-based services (LBS)

3.1.1 *Proofs-of-Presence*

In both the case of LBSs and in peer-to-peer scenarios, we can model the situation as follows: a *prover* P (e. g., an LBS client or peer device in a peer-to-peer setting) provides a *PoP* to a *verifier* V (a venue or a peer device) that they co-located, i. e., present in the same proximate environment.

Figure 3.1 shows an example of how PoPs can be used to verify the location check-ins of a client of an LBS. In this scenario, the client acts as the prover P and the venue at which the client wants to check in is the verifier V. First, the client provides a PoP to the venue (Step 1). After evaluating the PoP, the venue determines whether the client is co-located with it or not. If the client is determined to be co-located, the venue will sign a presence acknowledgement and return this to the client (Step 2). It can then use this acknowledgement to authenticate its presence at the venue when performing a location check-in with the LBS (Step 3) (In a variation of this scheme, the venue could also provide the presence acknowledgement directly to the LBS).

Prior works have proposed two main approaches for constructing PoPs: *beaconing-* (e. g., [121, 80]) and *context-based* (e. g., [143, 100]) PoPs, which we shall briefly describe in the following.

3.1.1.1 *Beaconing-Based PoPs*

In beaconing-based PoPs, verifier V emits a beacon signal b into its proximate environment, which prover P needs to capture with its on-board sensors (e.g., WiFi or Bluetooth). The beaconed information b is then either used directly by P as a proof of presence towards V, or, it is used in a proof-of-knowledge protocol between P and V to mutually establish their co-presence. The underlying assumption

with beaconing-based PoPs is that only a device actually co-located in the proximity of V will be able to accurately capture the beaconed information b. In peer-to-peer settings the beaconing-based approach has the drawback that verifier V inevitably has to reveal its presence in the context by emitting b which might not be desired in order to protect the privacy of verifier V. For example, if beaconing is performed by, e. g., the verifier emitting packets over WiFi or Bluetooth, the Bluetooth or WiFi MAC address of V will be exposed to anyone within wireless range of V.

3.1.1.2 *Context-Based PoPs*

In this work, we focus on context-based PoPs. These are based on V and P simultaneously sensing their ambient contextual environment. As in the previous chapter (cf. Sect 2.1.3), we define here *context* to mean any information about physical properties of the ambient surroundings that the devices involved can sense with the help of their on-board sensors like luminosity or audio sensors (i. e., microphones). The proof builds on the assumption that transient incidental fluctuations in the sensed contextual parameters can't be exactly sensed nor predicted by an external attacker $\mathcal{A}$ that is not located in the same context as V and P.

Earlier works have proposed a number of approaches for performing context-based PoPs [49, 100, 131, 140, 143]. In these approaches, the contextual measurements are either used directly to generate a shared key (e. g., [143]), or, prover P sends its measurements to V, that compares them to its own measurements. It often happens that due to sensing errors or timing jitter, measurements are not always identical between V and P. However, if the measurements are similar enough, they constitute a valid *context-based Proof-of-Presence (PoP)*.

3.1.2 *Context Guessing Attacks*

Earlier works on context-based PoPs have not considered what we in this work call *context guessing attacks*. In a context guessing attack, an adversary $\mathcal{A}$ that is *not* co-located with V uses information it has about V's context to *guess* probable parameter values that V is likely to observe. In related work, such attacks have either been considered to be out of scope [143, 140], or, they have simply assumed that the used context measurements b have sufficient entropy to resist guessing attacks [100]. However, in this work we show that for typical proximity sensing techniques like the use of WiFi or Bluetooth, context guessing is a relevant problem that needs to be appropriately taken into account in order to obtain proofs-of-presence that are resilient to guessing adversaries that have profiled information about the targeted context.

3.1.3 *Goals and Contributions*

We empirically analyse commonly used context modalities like Bluetooth and WiFi and show that the entropy of individual context observations needs to be taken into account to obtain reliable PoPs that can't be easily forged by a context-guessing adversary. To defend against such adversaries we propose two complementary approaches: *surprisal filtering* and utilising *longitudinal ambient context* observations for constructing resilient PoPs.

3.1.3.1 *Surprisal Filtering*

We develop the concept of surprisal filtering which is based on estimating the entropy of individual context measurements and filtering them in a way that only measurements having sufficient entropy are admitted as valid PoPs. The approach uses context profiling and a data mining algorithm for identifying frequent itemsets in order to estimate occurrence probabilities of particular context parameter combinations and use this information to rule out combinations that are easy to guess for the adversary.

3.1.3.2 *Longitudinal Ambient Context Observations*

Another approach for making context-based PoPs that are more resilient against guessing is to utilise longitudinal measurements of the ambient context like the luminosity or the audio environment. We show that by appropriately profiling such context modalities, sufficient inherent entropy from the context can be extracted to construct PoPs that are in most cases impractical to guess. This is due to the fact that contrary to earlier works (e. g., [49, 140]) that consider only momentary snapshots of the context, we monitor the context for a longer time and utilize short-term changes in the observed parameters to extract sufficient entropy for a reliable PoP.

3.2 BACKGROUND

A number of approaches for context-based PoPs have been proposed. In the following we will review relevant aspects of related work that we have adapted for our own approach.

3.2.1 *Use of WiFi for Proximity Verification*

Varshavsky *et al.* developed *Amigo* [143], which is a system for co-location verification of mobile devices. It is based on observing the Received Signal Strength Indications (RSSIs) of a number of WiFi packets originating from a WiFi access point that two co-located mobile devices can observe. If the RSSI values that both devices observe

are similar, they can be considered to be near to each other (i. e., 'co-located'), as correlations between fluctuations in the RF environment are locally limited. They will be observed in a similar way only by devices that are located close to each other. On the other hand, such fluctuations occur randomly, making them difficult to be guessed by an external adversary that is not in proximity. Using this approach for proofs-of-presence, however, requires both P and V to be located relatively close to each other, limiting the practicality of the proposed approach. For typical proof-of-presence applications, in which a mobile device user needs to prove their presence at a venue (e. g., a restaurant), or, in the same location as another user, where the peers are in the same room but not in immediate proximity, this approach is difficult to realise in practice.

Narayanan *et al.* [100] utilize the concept of *location tags* (first introduced by Qiu *et al.* [109]), i. e., small pieces of information that can be harvested from the ambient context. They construct protocols that utilise *private set intersection* [41] to allow two peers observing location tags to compare them in a way that does not leak information about their privately observed tags to the counterpart. Narayanan *et al.* discuss a number of different contextual modalities like GPS, Bluetooth, and GSM signals, audio and even local concentrations of atmospheric gases as potential sources of location tags. However, they evaluate only their WiFi broadcast packet-based location tag solution. Based on evaluation data originating from the WiFi network of a university campus, they estimate that approximately 10 bits of entropy could be harvested from distinct protocols of WiFi broadcast packets when using them as location tags.

Their approach has, however, the drawback that for harvesting location tags, both peers need to connect to the same WiFi access point. Access points using encryption protocols like WiFi Protected Access 2 (WPA2) [60] to protect their communications can't be used for harvesting location tags unless both peers are able to authenticate with it, limiting the practicality of the proposed approach. The ability to generate location tags is also heavily dependent on the traffic patterns of the used WiFi network. Low-traffic networks like residential home WiFi networks with only few users may display at times so low traffic rates that obtaining a sufficient number of location tags in a reasonable amount of time may be challenging.

3.2.2 *Multi-Modal Proximity Verification*

Our work is inspired by earlier proximity verification approaches presented by Truong *et al.* [140] and Shrestha *et al.* [131]. They utilise a number of different contextual parameters obtained by monitoring the context to verify the co-location of two trusted devices like the smartphone and laptop computer of a user, or, the smartphone and

an automated teller machine (ATM). These approaches focus on the problem of *relay attacks* in which adversary $\mathcal{A}$ uses a 'ghost-and-leech' set-up [69] to relay messages of the proximity verification protocol between prover P and verifier V that are *not* co-located, over a fast long-distance link. The goal of $\mathcal{A}$ is to make V believe that P is in its vicinity even though it is not. This scenario is relevant to use cases utilising *zero-interaction authentication*, e. g., in key-less entry systems for cars, or, contactless payment cards.

In the scenarios they consider, however, they assume that both peers performing the proximity verification are trusted. In our proof-of-presence scenario, however, the scenario is different. We need to take into account the possibility that the adversary $\mathcal{A}$ may assume the role of the prover in the proximity verification protocol with verifier V. In contrast to the zero-interaction authentication use case, we must therefore also take into account the entropy of the proof-of-presence in order to make them resilient against guessing attacks, as discussed in Sect. 3.1.2.

3.3 CONTEXT-BASED PROOFS-OF-PRESENCE

The basic concept of context-based PoPs is shown in Fig. 3.2. Prover P initiates the process by placing a `PoP_REQ` request along with a timestamp t to synchronise on to verifier V. Both prover P and verifier V then record at time point t context measurements $C_P(t)$ and $C_V(t)$, respectively. Prover P sends its measurement $C_P(t)$ to V, which compares it to its own measurement $C_V(t)$. If P's measurement is similar enough to its own, i. e., if

$$dist(C_V(t), C_P(t)) \leq \Delta_{thr}, \tag{3.1}$$

verifier V will accept P's PoP. Here, $dist(\cdot, \cdot)$ denotes a suitable distance function used to determine the similarity between context measurements and Δ_{thr} a predefined similarity threshold determining how much deviation between measurements is acceptable for them still to be considered coming from the same context.

In practice, we will implement the context measurement comparison with the help of a binary classifier that is trained to distinguish between co-located and non-co-located context measurements. For training this classifier, we use a number of context features based on the to-be-compared context measurements.

3.3.1 *Context Features*

Earlier works have investigated a number of different context features in various modalities for constructing context-based PoPs. Used modalities include, e. g., audio [49, 140], luminosity [49], concentrations

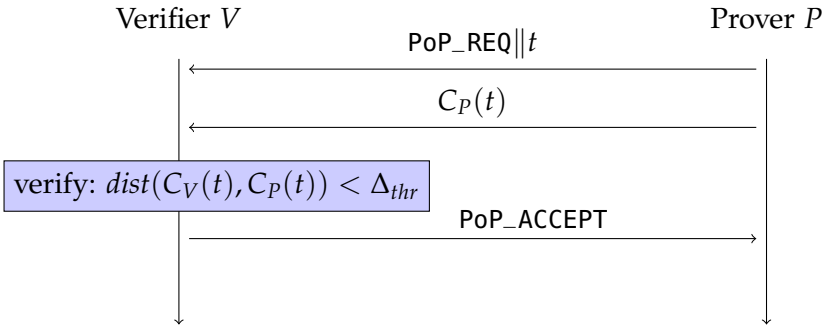


Figure 3.2: Context-based proof-of-presence

Table 3.1: Features used for context-based PoPs

	FEATURE NAME	DEFINITION
f_1	Jaccard distance	$1 - \frac{\|C_V \cap C_P\|}{\|C_V \cup C_P\|}$
f_2	Hamming distance	$\frac{\sum_{i=1}^{n} \|m_i^P - m_i^V\|}{n}$
f_3	Euclidean distance	$\sqrt{\sum_{i=1}^{n} (m_i^P - m_i^V)^2}$
f_4	Exponential of difference	$\frac{\sum_{i=1}^{n} e^{\|m_i^P - m_i^V\|}}{n}$
f_5	Squared rank differences	$\sum_{i=1}^{\|C_V \cap C_P\|} (rank(m_i^P) - rank(m_i^V))^2$

$m_i^V \in C_V, m_i^P \in C_P$: signal strength measurements of individual elements (WiFi APs or Bluetooth devices) in the context measurements C_V and C_P of the verifier V and prover P, respectively.
$rank(m_i^V), rank(m_i^P)$: rank of m_i^V or m_i^P in C_V or C_P, respectively, sorted in ascending order.

of atmospheric gases, ambient temperature, humidity and air pressure [131], WiFi [143], Bluetooth and GPS [140]. As Truong *et al.* [140] have concluded that Bluetooth and WiFi are effective modalities for constructing context-based PoPs, and these are readily available on contemporary mobile devices, we will focus on these modalities for constructing context-based PoPs and adopt five of the features used by Truong *et al.* which are based on the context measurements C_P and C_V of prover P and verifier V, respectively, as shown in Tab. 3.1.

3.4 CONTEXT GUESSING ATTACKS

Our main focus is on developing countermeasures against *context-guessing attacks* in which an adversary $\mathcal{A}$ attempts to fake its presence in the proximity of a verifier V. In the following we will describe the attack scenario and our assumptions about the adversary in detail.

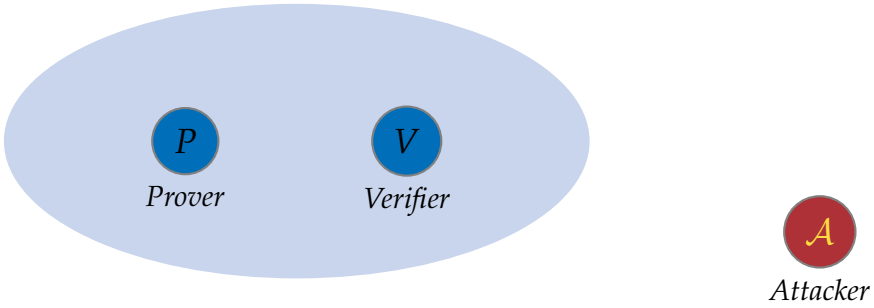


Figure 3.3: Adversary model

3.4.1 *Adversary Model*

As discussed in Sect. 3.1, our adversarial model is motivated by two scenarios in which PoPs are applied: peer-to-peer applications utilizing location information of users, and, location-based services.

We model both scenarios as shown in Fig. 3.3: A *prover* P wants to prove to a *verifier* V that it is co-located with it, i. e., in the proximity of V, by presenting a *Proof-of-Presence (PoP)* which V will evaluate to determine whether the PoP is valid or not. The adversary $\mathcal{A}$ is a node *not* in the proximity of V. The goal of $\mathcal{A}$ is to fabricate a fake PoP which V will erroneously accept as a valid proof-of-presence.

3.4.1.1 *Adversary in Peer-to-Peer Use Case*

An exemplary peer-to-peer use case is provided by a friend finder app that notifies users if they are located in the proximity of their friends as determined by the location check-ins of users. Adversary $\mathcal{A}$ is an intrusive user of such an application who wants to engage in cyber stalking of other users by doing fake location check-ins in a number of different places in order to learn the presence of other users in those locations.

3.4.1.2 *Adversary in Location-Based Services Use Case*

In the LBS scenario, adversary $\mathcal{A}$ is a malicious user of the the LBS that wants to gain unjustified benefits like discounts or rebate cards at specific venues like restaurants or shops by performing numerous fake location check-ins in those venues, even though $\mathcal{A}$ hasn't visited the locations in reality.

3.4.2 *Context Guessing*

As discussed in Sect. 3.1.2, a strategy of adversary $\mathcal{A}$ can be to try to fabricate an adversarial measurement $C^*_{\mathcal{A}}$ that is similar enough with the measurement $C_V(t)$ of the verifier at the current time point t, i. e.,

$dist(C_V(t), C^*_{\mathcal{A}}) \leq \Delta_{thr}$, so that verifier V will be fooled into accepting it as a valid PoP.

EXAMPLE. Considering the LBS use case, adversary $\mathcal{A}$ could be a user visiting a venue V. During its visit at the venue, at time point t, $\mathcal{A}$ will perform a context measurement $C^*_{\mathcal{A}}(t)$. Later, at time point $t + k$, adversary $\mathcal{A}$ can use this measurement as a proof of presence to try to convince venue V—now acting as a verifier—that it is co-located with it. If the context information has not changed significantly since when $\mathcal{A}$ was visiting the venue, i. e., if $dist(C_V(t + k), C^*_{\mathcal{A}}(t)) \leq \Delta_{thr}$, V will accept $\mathcal{A}$'s measurement erroneously as a genuine PoP and issue to it an acknowledgement of $\mathcal{A}$'s presence at V at time point $t + k$. ■

3.4.3 *Susceptibility of PoPs to Context-Guessing Attacks*

To evaluate the susceptibility of context-based PoPs to such context-guessing attacks, we simulate an adversary $\mathcal{A}$ that uses context measurements $C_{\mathcal{A}}$ it has obtained earlier from the context of verifier V as PoPs towards the verifier. We evaluate the effectiveness of the attack in terms of *false positive rate (FPR)*, i. e., the rate at which adversarial context measurements $C_{\mathcal{A}}$ that were made at the *same location but not at the same time* as the verifier's measurement C_V are erroneously accepted by V as genuine PoPs. We selected this attack model, as it is one of the simplest and most straightforward strategies that an adversary might play. Nevertheless it can be used to demonstrate the basic problem that context guessing attacks in general represent.

3.4.3.1 *Evaluation Datasets*

ZIA DATASET. The authors of [140] kindly provided us their dataset for evaluating the possibility of context guessing attacks, enabling us to directly compare our approach with their results. Note, however, that their use case is concerned with zero-interaction authentication (ZIA) and the adversary model is therefore different from ours. Their scenario is concerned with protection against relay attacks and both prover P and verifier V are assumed to be trusted, whereas we investigate the possibility of context-guessing attacks, where attacker $\mathcal{A}$ takes the role of a malicious prover. Nevertheless their dataset is very useful in examining the susceptibility of context-based PoPs to context-guessing attacks, as their dataset contains samples of context measurements of co-located and non-co-located devices.

The (benign) ZIA dataset contains measurements of visible WiFi access points and Bluetooth devices and their signal strengths collected simultaneously by two different smartphones. The dataset contains in total 2302 such context sample pairs. Out of these samples 1140 measurement pairs are from co-located devices, while in 1162

samples, the measuring devices were not-co-located when the context measurements were taken.

Based on the benign ZIA dataset we constructed an *attack dataset* simulating context guessing attacks by re-mapping context measurements. The re-mapping of measurements was done by combining pairs of context measurements that were recorded in the same location, but *at clearly different times*, i. e., 6 to 24 hour apart. The ZIA dataset contains ground truth labels for measurement pairs indicating whether the measurement devices were co-located or not, but no information about where the measurements were actually performed. Therefore we utilised the set of observed WiFi access points contained in each measurement as an indicator for the location in which the measurement was done. To determine a criterion on which to determine whether measurements at different times were performed in the same location or not, we compared co-located measurement pairs to non-co-located measurements in the benign ZIA dataset. We calculated the Jaccard distance (feature f_1 in Tab. 3.1) for the sets of observed WiFi access points for these measurement pairs, and could observe that a Jaccard distance threshold of 0.9 provided good separation between co-located and non-co-located measurement pairs.

We therefore selected this threshold as a criterion for determining locations and assumed that any measurements for which the Jaccard distance of the associated sets of observed WiFi access points is below 0.9 have been made at the same location. Using this decision criterion we then paired all measurements with other measurements that were made in the same location but at a different time.

CONXPOP DATASET. In addition to the ZIA dataset, we also collected an additional dataset that incorporated also other contextual features than Bluetooth and WiFi measurements. The data collection was done with the help of a data collection application that was given to a number of test users who installed the application onto their Android smartphones. The app continuously measured contextual parameters and periodically uploaded the measurements onto our data collection server for off-line analysis. Tab. 3.2 shows the context parameters collected by the data collection app.

The test users collecting the ConXPoP dataset included volunteered members of the research lab sharing nearby offices and visiting the same lunchtime restaurants. This allowed the test users to aggregate a rich collection of co-located context measurements related to normal everyday situations. All participants were provided in writing an explanation about purpose and goals of the data collection experiment and a description of the context parameters collected by the data collection app. Participants were free to stop or interrupt data collection at any point in time by disabling the data collection app.

Table 3.2: Measured context parameters in the ConXPoP dataset

PARAM	DESCRIPTION	SAMPLING FREQUENCY
1	WiFi AP MAC addresses	$1/\text{min}$
2	WiFi AP RSSIs	$1/\text{min}$
3	BT device MAC addresses	$1/\text{min}$
4	BT device RSSIs	$1/\text{min}$
5	Ambient noise level	continuous
6	Ambient luminosity	continuous

Participants had also the possibility to revoke their participation in the experiment at any time and demand their data to be deleted.

In the process of the data collection all participants were asked to provide through the UI of the data collector app information about the contexts they often visited (e. g., home, office, lunch restaurant, etc.) and to indicate, which other test participant devices were co-located with the user's own context collector device at a particular point in time. Devices of other test participants were hereby identified with easily recognisable nicknames. Participants were asked to mark only such devices to be co-located with the participant's device that were likely to remain in proximity for the following two minutes, so that a significant fraction of the subsequent noise level and luminosity measurements would be from co-located devices.

In order to obtain context measurements from such contexts in which test participants typically are alone, each participant used two context measurement devices: their primary smartphone and an "alter ego" device. By taking the alter ego device also to such contexts that no other test participant visited, test participants were able to provide context measurement pairs also from such contexts.

During a data collection period of 10 days, participants provided a total of 5602 labelled co-located context measurement pairs. Using these data, we constructed for each participant a benign and an attack dataset. Each benign dataset was constructed by pairing context measurements that were marked as being co-located by the participant or some other participant. A roughly equal amount of measurement pairs were generated by pairing measurements that were not marked to be co-located.

The attack dataset was generated by letting each participant at a time act as verifier V. Each context observation $C_V(t)$ of the participant originating from a specific frequently-visited target context $X \in \{\text{"Home"}, \text{"Office"}, \text{"Restaurant"}\}$ was paired with potential adversarial context observations $C_{\mathcal{A}}(t-k)$ made in the same context X, where all participants were allowed to assume the role of $\mathcal{A}$, and the value of k was varied between 6 to 24 hours.

Table 3.3: Co-location classifier FPR on benign datasets

DATASET	BLUETOOTH FEATURES	WIFI FEATURES
ZIA	2.5%	1.6%
ConXPoP	14.2%	11.0%

3.4.3.2 *Evaluation Results*

To evaluate the effectiveness of the context guessing attack, we first evaluated the performance of benign co-location classification in both datasets and then compared this to the performance of the classification for the attack datasets. We used false positive rate (FPR) as the measure of fitness, as it is an indicator of the adversary's ability to fabricate PoPs that will be erroneously accepted as valid proofs-of-presence. We used Multiboost as the classification algorithm using J48Graft as the base learner and used the Weka data mining suite [50] to evaluate our experiments.

BENIGN CO-LOCATION CLASSIFICATION The samples in the ZIA dataset were used to train a classifier for verifier V, using both co-located and non-co-located measurement pairs. We evaluated the classifier's performance on the benign dataset using 10-fold cross-validation. As shown in Tab. 3.3, we were able to corroborate the results of Truong *et al.* [140], as we obtained an FPR of 2.5% for Bluetooth features and and 1.6% for WiFi features of the ZIA dataset.

For the ConXPoP dataset, the performance was somewhat worse, as the FPR was 14.2% for Bluetooth features and 11.0% for WiFi-based features. Combining both features brings the FPR down to 9.3%. This difference in performance is due to the more challenging experimental set-up in the ConXPoP experiment. Whereas in the ZIA dataset co-located and non-co-located measurements can be more clearly separated, the ConXPoP data collection set-up was unfortunately more ambiguous. The criterion for co-location was that any devices located in the same room were considered to be co-located, and other devices not. This criterion was selected in order to allow test users to visually observe which other participants and their devices they could see present in the same room at a particular point in time. However, as the test participants were office mates using rooms located next to each other, their devices partially shared the same WiFi and Bluetooth environment, even though they were not co-located according to the above criterion. This ambiguity makes it more difficult for the classifier to clearly distinguish between co-located and non-co-located measurements, resulting in a higher false positive rate.

Table 3.4: FPR of the co-location classifier in view of context-guessing attacks

DATASET	BLUETOOTH	WIFI	BLUETOOTH + WIFI
ZIA attack	35.1%		
ConXPoP attack	21.9%	26.0%	23.5%

CONTEXT GUESSING SUCCESS To evaluate the performance of the classifier in view of a context guessing attack, we trained the co-location classifier using the benign datasets as training data and the attack datasets for testing. The results are shown in Tab. 3.4

As can be seen, the FPR of the co-location classifier significantly deteriorates when confronted with adversarial context measurements related to context replay attacks. For both attack datasets, the false positive rate lies between roughly 22 and 35 per cent, meaning that the adversary would have a chance of at least one out of five in succeeding in a context-guessing attack. This shows clearly that in a scenario in which the prover P can't be fully trusted by the verifier V, context measurements alone do not guarantee reliable proofs-of-presence. In practice this means that verifier V also needs to take into account the risk of possible context-guessing attacks when assessing the validity of context-based PoPs.

3.5 HARDENING CONTEXT-BASED PROOFS-OF-PRESENCE

In order to defend context-based PoPs against context-guessing attacks, we introduce two countermeasures, *surprisal filtering*, and, increasing the entropy of PoPs by using *longitudinal ambient context* modalities.

The first countermeasure, surprisal filtering, discussed in Sect. 3.5.1, aims at identifying and filtering out such PoPs that are easy to guess for the adversary. This is done by estimating the entropy of individual PoPs and dismissing such PoPs that would be too easy to guess. This estimate is based on the notion of *surprisal*, i. e., the self-information associated with the context observation $C_V(t)$ of the verifier at time point t. The notion of surprisal is closely related to entropy, however, with a slight difference. Whereas entropy refers to the *average* uncertainty associated with a random variable, the term *surprisal* refers to the uncertainty associated with a *particular outcome* of the random variable.

The second countermeasure aims at increasing the entropy of context-based PoPs inasmuch as to make successful context guessing impractical for the adversary. In contrast to earlier context-based co-location verification methods (e. g., [49, 124, 140, 143]) that use only

momentary snapshots of the context, our approach uses a longitudinal approach, in which we monitor the ambient properties over a longer time and use observed changes in values like luminosity or sound energy level to extract sufficient entropy from the context. This approach is described in Sect. 3.5.2.

3.5.1 *Surprisal Filtering*

Surprisal filtering is based on estimating how likely it is that adversary $\mathcal{A}$ could successfully fabricate an adversarial measurement C_A^* that is similar enough to verifier V's context measurement C_V to be accepted as genuine. The estimate is based on profiling each of V's contexts and using these profiles to estimate the occurrence probability of a measurement C_V in a context X. Our intuition is that if the occurrence probability of C_V in X is low, it is also more difficult for the adversary to fabricate a measurement that is similar with it, even if $\mathcal{A}$ has monitored context X earlier. Based on the probability estimates of each context-based PoP, verifier V can reject any proofs for which the occurrence probability is high, i. e., that have a significant risk of being fabricated.

Formally, we consider an observable context X of verifier V as a random variable O_X taking concrete context measurements C_V as its value. The surprisal of a context measurements C_V is therefore a measure of the uncertainty of that particular outcome in view of the distribution of O_X. In practice, even in the most favourable case for adversary $\mathcal{A}$, it will have only partial information about the distribution of O_X, since according to our adversary model, the adversary will not be permanently present in the verifier's context X. However, since V does not know how much $\mathcal{A}$ knows about the distribution of O_X, we make the conservative assumption that $\mathcal{A}$ has the same information about the distribution of O_X as V.

Definition 14 (Surprisal filtering) *We define* surprisal filtering *to be a function $\varsigma : \mathcal{C} \times \mathcal{X} \to \{accept, reject\}$ where $\mathcal{C}$ denotes the domain of context measurements and $\mathcal{X}$ the set of V's known contexts. The surprisal filtering function ς maps a context measurement C_V observed by V in context $X \in \mathcal{X}$ to an acceptance decision* accept *or* reject *based on the surprisal value $I_X(C)$ as follows*

$$\varsigma(C_V, X) = \begin{cases} accept, & \text{if } I_X(C_V) \geq I_{thr} \\ reject, & otherwise \end{cases} \tag{3.2}$$

The evaluation of the surprisal value $I_X(C_V)$ is described in detail in Sect. 3.5.1.1. The rationale for using it for hardening PoPs is that static context information like, e. g., the observable link layer addresses of WiFi access points in an office, are not likely to change very quickly over time. Therefore an adversary $\mathcal{A}$ who has visited

the target context earlier can use his earlier measurements of the WiFi environment to fabricate context measurements even when it is not located in the target context. Dynamic context information, on the other hand, like the Bluetooth Medium Access Control (MAC) addresses of the smartphones of customers in a shop are likely to be much more volatile over time and therefore a much better basis for reliable context-based PoPs, as they are much harder to predict. In the following we show how the notion of *surprisal* can be used to measure the dynamicity of context information in a specific context and how it can be used to harden context-based PoPs to be more difficult to predict by the adversary.

3.5.1.1 *Surprisal of Context Measurements*

To identify context measurements C that are at risk of being easy to guess by adversary $\mathcal{A}$, we measure the how difficult it would be for it to guess it given the history of context observations $\mathcal{H}_X$ in context X. As discussed above in Sect. 3.5.1, we assume a strong adversary who has the same access to the context history data of X as verifier V. We have therefore to assume that $\mathcal{A}$ can use the full history $\mathcal{H}_X$ to try to fabricate context measurements $C_{\mathcal{A}}$ that are likely to be observed in X. As discussed in 3.5.1, we model the occurrence of context measurements in context X with the random variable O_X. The probability that a particular context measurement C is observed in X is therefore $P(O_X = C)$. The *surprisal* of measurement C is the *self-information* of this outcome.

Definition 15 *The* surprisal $I_X(C)$ *of a context measurement* C *in context* X *is the* self-information *of this measurement in view of the* context history $\mathcal{H}_X$ *of the context. Formally, it is defined as*

$$I_X(C) = \log_2\left(\frac{1}{P(O_X = C)}\right) = -\log_2\left(P(O_X = C)\right) \tag{3.3}$$

and is measured in bits.

In general, to calculate the probability of a context measurement C in context X, we adopt a frequentist interpretation of probability and calculate $P(O_X = C)$ as the fraction of times that C has been observed in X. Given a context measurement $C = \{d_1, d_2, \ldots, d_n\}$ in context X, its occurrence probability is therefore calculated as

$$P(O_X = C) = \frac{\|\{C_i \in \mathcal{H}_X | C \subseteq C_i\}\|}{\|\mathcal{H}_X\|} \tag{3.4}$$

EXAMPLE. Let us consider the surprisal of Bluetooth device measurements. Let us assume the context history database $\mathcal{H}_X$ of context X contains a total of $n = 100$ context measurements. In this history, device A has been observed a total of 55 times and device B has been

observed in 35 measurements. Out of these measurements, 15 are such that both A and B occur in the same measurement. For observations containing individual measurements we obtain following probability estimates: $P(O_X = \{A\}) = \frac{55}{100} = 0.55$ and $P(O_X = \{B\}) = \frac{35}{100} = 0.35$. For a measurement containing both devices the estimate is $P(O_X = \{A, B\}) = \frac{15}{100} = 0.15$. The corresponding surprisal values are correspondingly: $I_X(\{A\}) = -\log_2(0.55) \approx 0.86$ bits, $I_X(\{B\}) = -\log_2(0.35) \approx 1.51$ bits and $I_X(\{A, B\}) = -\log_2(0.15) \approx 2.74$ bits. ■

To calculate the occurrence probability of a context measurement $C = \{d_1, d_2, \ldots, d_n\}$ in context X in practice, we need to calculate how many times that combination occurs in the context history $\mathcal{H}_X$. In case it is a frequently occurring combination, we need to reject it as a context-based PoP, as the surprisal of the measurement is low. Identifying frequently occurring element combinations in databases is a problem that has been extensively studied in data mining literature in the context of *frequent itemset* mining. One of the most well-known algorithms for this task is *Apriori* [2], which, given a frequency threshold, finds all frequent itemsets, i. e., combinations of items occurring more often than the given frequency threshold in the database along with their occurrence counts. We therefore utilize the Apriori algorithm to find any frequent WiFi access point combinations in the context history $\mathcal{H}_X$.

3.5.2 *Longitudinal Ambient Modalities*

In some contexts like the home of the user the WiFi and Bluetooth environment tend to be rather static, as the set of devices in such contexts typically doesn't change much. This means that the WiFi and Bluetooth environments can't be used for reliable context-based PoPs, as the context would be relatively easy to guess for the adversary.

To be able to do reliable context-based PoPs also in contexts in which the entropy of the WiFi and Bluetooth environments is low, we introduce a complementary approach for generating PoPs that is based on longitudinal monitoring of ambient physical context modalities like luminosity and audio.

3.5.2.1 *Ambient Light*

Modern smartphones are nowadays typically equipped with luminosity sensors, e. g., for the purpose of adjusting the device's screen brightness according to the surrounding lighting conditions. Obtaining luminosity measurements is therefore easy and, as the luminosity sensor doesn't consume much energy, continuous tracking of the ambient luminosity is feasible.

Halevi *et al.* have investigated the use of ambient light for co-presence verification [49]. However, their approach merely compares the average luminosity level of a short context snapshot to determine

whether the two devices performing the measurement are co-located or not. If the average luminosity measurements do not deviate much, the devices are deemed to be co-located. This scheme is therefore easy to overcome by a malicious prover $\mathcal{A}$ who has the opportunity to profile the context of verifier V beforehand. It just needs to fabricate a context measurement by replaying the observed average luminosity value to the verifier in order to succeed with high likelihood.

We therefore adopt a more sophisticated approach. Instead of utilising only momentary snapshots, we monitor the ambient luminosity continuously over a slightly longer time period, e. g., one minute, and derive a *context fingerprint* from the relative changes of the luminosity over time. Our intuition is that such changes often are caused by random events like human activity in the context and are therefore very difficult to predict by adversary $\mathcal{A}$, making them an amenable source of entropy for context-based PoPs.

3.5.2.2 *Audio*

A number of approaches have been investigated for using ambient audio for co-location verification. Halevi *et al.* [49] used time- and time-frequency based similarity measures between audio samples for co-location verification, whereas Truong *et al.* [140] used 10-second samples with similar similarity measures. We, however, we take a slightly different approach and sample the ambient acoustic environment for a longer period, i. e., one minute, and monitor for changes in the ambient sound level. Also here our intuition is that such changes are caused by, e. g., user actions that are difficult to predict by the adversary, thus providing more robustness against guessing attacks.

3.5.2.3 *Ambient Context Sampling*

To construct the PoP based on the above ambient context modalities, prover P and verifier V will monitor sequences of context measurements $M = (m_1, m_2, \ldots, m_n)$, where each measurement m_i is the average context parameter reading during a time window w. In practice, we propose to use one-minute (i. e., $n = 60$) measurement sequences with windows of $w = 1\,\text{s}$.

We chose one minute as the measurement length, as we think this to be a sufficiently long time period to capture enough changes in the environment for effective PoPs, while at the same time keeping the time required for the PoPs short enough to be practical. A one-minute delay should not represent a problem in most application scenarios, as in many cases the context sampling can happen in the background. For example, PoPs for location check-ins in OSNs can be performed in the background after the user has checked in, without the user having to wait for it to complete. Only in the case that the PoP fails

Table 3.5: Additional feature for luminosity and audio measurements

FEATURE	FEATURE NAME	DEFINITION
f_6	Maximum cross-correlation	$\underset{\tau}{\operatorname{argmax}} R_{VP}(\tau)$

Where $R_{VP}(\tau)$ denotes the cross-correlation between M_V and M_P at displacement τ

or is rejected might the user receive an error message notifying him about it.

The placement of devices in the context may have a significant impact on the signal levels of audio and luminosity measurements that the devices record. Our scheme can, however cope with this as it focuses on the observable *changes* in the readings and not on absolute observed signal values. Before processing the measurement sequences, we therefore apply min-max scaling on them so that all measurements in the sequence assume values between 0 and 100.

3.5.2.4 *Ambient Context Features*

Similar to the WiFi and Bluetooth-based context modalities, we calculate also the mean Hamming distance, the Euclidean distance, the mean exponential of difference and the sum of squared rank differences between the measurements M_V and M_P of verifier V and prover P as features for co-location classification. These correspond to features f_2 to f_5 in Tab. 3.1. In addition to these features we introduce also maximum cross-correlation shown in Tab. 3.5 as a feature for luminosity and audio measurements.

3.6 EVALUATION

To evaluate the performance of our hardening approaches, we apply our hardening measures on the collected datasets and evaluate the changes performance of the context-based PoPs in terms of false positive rate (FPR) and false negative rate (FNR). Here, false negative rate refers to the rate at which co-located measurements will be rejected by the system as valid PoPs.

3.6.1 *Performance of Surprisal Filtering*

To evaluate the effectiveness of surprisal filtering, we used the *Apriori* algorithm to identify the frequently occurring combinations of observed Bluetooth or WiFi devices in various contexts in the ConXPoP benign dataset. As frequency thresholds we used values corresponding to surprisal values $I_{thr} = 2$ bits and $I_{thr} = 4$ bits. We then applied

Table 3.6: Improvement in FPR when surprisal filtering is applied on the ConXPoP attack datasets

		FPR IMPROVEMENT FOR $I_{thr} = n$ BITS			
		$n = 2$		$n = 4$	
USER	UNFILTERED FPR	BT	WIFI	BT	WIFI
A	13.0 %	−6.1 %	−2.0 %	−8.9 %	−2.8 %
B	37.8 %	−27.2 %	−5.4 %	−31.1 %	−5.6 %
C	37.2 %	−0.3 %	−4.9 %	−0.3 %	−5.3 %
D	21.4 %	−17.4 %	0.0 %	−19.4 %	0.0 %
E	16.2 %	−11.6 %	−7.6 %	−13.8 %	−10.0 %
F	40.5 %	−23.8 %	−7.7 %	−26.8 %	−9.5 %
Avg	27.7 %	−14.4 %	−4.6 %	−16.7 %	−5.5 %
Relative change		−52.0 %	−16.6 %	−60.4 %	−20.0 %

the identified frequent device combinations on the ConXPoP attack dataset to reject any PoPs with insufficient surprisal value and observed the resulting FPR. The results are shown in Tab. 3.6.

Using surprisal filtering for PoPs reduces the FPR for Bluetooth-based context measurements by 52 to 60 per cent and by 16 to 20 per cent for WiFi-based PoPs, significantly reducing the adversary's odds for successful context-guessing attacks.

3.6.1.1 *PoPs Utilising Longitudinal Ambient Context*

For evaluating context-based PoPs using the longitudinal ambient context, we utilised the luminosity and audio measurements included in the ConXPoP dataset (the ZIA dataset did not contain measurements in these context modalities). We extracted for these measurements features f_2 to f_6 and augmented the WiFi and Bluetooth-based features of the basic PoP scheme with them for training hardened co-location classifiers. We experimented with different combinations of feature sets to assess their impact on classification performance in terms of FPR and FNR. The results are shown in Tab. 3.7.

As can be seen from the results, augmenting the WiFi and Bluetooth-based context features with longitudinal features based on ambient audio and luminosity significantly reduces both FPR and FNR. When comparing to a combination of Bluetooth and WiFi features, adding luminosity and audio-based features reduces the FPR by more than a half from 9.3% to 4.2%. Also the probability of false rejections, i. e., the FNR goes down from 6.2% to 2.4%. The inclusion of longit-

Table 3.7: Performance of PoPs utilizing audio and luminosity modalities

	BENIGN		ATTACK	
CLASSIFIER FEATURES	FPR	FNR	FPR	FNR
Luminosity	20.1 %	14.3 %	1.1 %	0.0 %
Audio	19.2 %	16.0 %	0.4 %	0.0 %
Luminosity+Audio	9.3 %	9.2 %	0.4 %	0.0 %
BT	16.1 %	9.8 %	21.9 %	0.0 %
WiFi	11.0 %	9.9 %	26.0 %	0.0 %
BT + WiFi	9.3 %	6.4 %	23.5 %	0.0 %
Luminosity+Audio+BT+WiFi	4.2 %	2.4 %	3.6 %	0.0 %

udinal ambient context features provides therefore a more than 50% improvement in both the security (less falsely accepted PoPs) and usability (less falsely rejected PoPs).

3.7 DISCUSSION

As our results in Sect. 3.1.2 show, *context guessing* is a serious problem for context-based PoPs in scenarios in which the prover can't be trusted, as an adversary $\mathcal{A}$ can utilise profiled information it has possibly obtained earlier about the context of verifier V to fabricate context measurements and present these as its PoP towards V. Depending on the context, modalities like Bluetooth and WiFi are especially prone to such attacks. However, by profiling the relevant contexts of the verifier V it can evaluate the *surprisal* of each PoP in view of the history of context measurements in the particular context. Surprisal is a measure for estimating how easy it would be for the adversary to guess the context measurement presented as a PoP based on the context history. Since our adversary model assumes a strong attacker (cf. Sect. 3.4.1), we have to assume that it has similar access to the context history as the verifier. Based on the surprisal value, verifier V can reject such PoPs that would be too easy to guess by $\mathcal{A}$.

3.7.1 *Impact of Context Entropy*

In some contexts like the home of a user, the WiFi and Bluetooth environments can be relatively static, as the set of observable WiFi access points and Bluetooth devices typically does not change much over time, because the set of persons and consequently their Bluetooth devices present in the context stays more or less the same. This can have the effect that there is not much entropy in WiFi and Bluetooth

measurements, making it difficult to construct PoPs with sufficient surprisal to be accepted as valid.

However, especially in LBS-related scenarios (cf. Sect. 3.4.1) the main target contexts are public venues like shops and restaurants that typically contain a lot of entropy, as the persons present in the context are constantly changing and mostly unpredictable. In contexts in which the surprisal of PoPs in a particular modality is not sufficient, the PoPs can be augmented by including also features based on longitudinal ambient context modalities like luminosity and audio in the PoPs. As the results in Sect. 3.6 show, this significantly helps to improve both the resilience of PoPs against context guessing attacks, and, helps to improve the accuracy and usability of our scheme by reducing the rate of falsely rejected PoPs.

3.7.2 *Privacy Considerations*

In contrast to earlier works that utilise audio measurements for co-presence verification [49, 140], our approach provides privacy advantages, as it does not require prover P to send actual fine-grained audio data to the verifier, but uses only the average noise level and its changes in constructing the PoP. This reduces significantly the risk of possible (unintended) exposure of sensitive information.

3.7.2.1 *Limitations*

A fundamental problem for all context-based co-location verification schemes is given by so-called relay attacks, in which a remote adversary $\mathcal{A}$ collaborates with an accomplice that is located in the proximity of verifier V and who forwards actual measurements of V's context to $\mathcal{A}$ to be used as context-based PoPs. To the best of our knowledge, only distance bounding-based techniques (e.g., [55]) can provide effective protection against such attacks. The drawback of distance-bounding approaches is however, that they require dedicated special hardware capabilities that are typically not available on regular smartphones.

However, in our application scenarios relay attacks would not seem to play a significant threat, as it would be prohibitively complex and costly for an adversary to deploy accomplices in all possible contexts that a targeted user could visit. The threat would therefore be limited to a few specific contexts. However, also in these cases the use of several different longitudinal context modalities would make implementation and execution of the attack more complex as it is not any more sufficient to merely relay protocol messages between the verifier V and the adversary $\mathcal{A}$, but accomplices would need to actively participate in sensing the context of the verifier in all used context modalities.

3.8 RELATED WORK

As discussed in Sect. 3.2.2, our work is closely related to the works of Truong *et al.* [140] and Shrestha *et al.* [131]. These works focus, however, on a different scenario, i. e., co-location verification methods for mitigating external relay attacks against proof-of-presence protocols in a zero-interaction authentication setting. They utilise a wide range of different context modalities like WiFi, Bluetooth, ambient audio and luminosity, and evaluate their suitability for verifying co-presence of devices.

In their scenario, however, both peers can assumed to be benign and trusted. This rules out the need to consider context-guessing attacks. This is fundamentally different in our scenario, in which the prover can be malicious and therefore engage in context-guessing attacks. In fact, our work shows that such attacks play a significant role in the security of context-based PoPs.

3.8.1 *Beaconing-Based approaches*

Several works have proposed approaches in which location proofs are built based on information that is beaconed into the proximity of a venue using a distance-limited protocol that is observable only in the vicinity of the venue. One of the first works proposing this approach was presented by Saroiu and Wolman *et al.* [121]. They discuss six scenarios concerning LBSs in which users of the LBS may have incentives to engage in location cheating. To address this they propose a system in which location proofs are based on beaconing information over the SSID of a dedicated AP installed at the venue. The rationale is that only devices within wireless range of the access point are able to receive these beaconed signals. A drawback of their scheme is that it requires the installation of such dedicated APs and is therefore applicable only in venues where such APs are available. Generic mobile scenarios related to, e. g., OSNs can't therefore be covered by their scheme.

Another approach for verifying device co-location is the SMILE framework by Manweiler *et al.* [80], which is based on a scheme in which mobile devices beacon cryptographic keys into their proximity and simultaneously record keys beaconed by other devices. This beaconing is done, e. g., over Bluetooth or some other proximity communication protocol. The SMILE framework consequently allows users to establish proofs of co-location after the fact by comparing the sets of keys with the help of a third-party server. Their scheme has, however the major security drawback that individual users' devices are required to continuously broadcast their keys into their environment, thus making them potentially traceable over all the contexts they visit.

Carbunar *et al.* [22] present a scheme for privacy-preserving check-ins in so-called *Geo-Social Networks*, i. e., OSNs explicitly involving the physical location of users. Their construction is called *GeoBadges*, and it utilises mix networks and a protocol involving blind signatures to provide anonymous proofs of repeated visits to specific venues. Their scheme relies on dedicated hardware installed at the venue to display, e. g., changing Quick Response (QR) codes which users need to scan with their smartphones in order to construct the proof of presence at the venue at a particular point in time.

Polakis *et al.* [108] present a similar scheme for verifying check-ins at particular venues. Their scheme is based on temporary codes that can be verified by the LBS. These codes need to be scanned by the client devices using dynamic NFC tags which the user needs to scan with his device. In this scheme, location-verification is based on the transferral of the code over NFC, as its range is limited to a very short distance. However, also this implementation requires the use of dedicated hardware and is therefore not applicable in generic mobile scenarios.

3.8.2 *Proofs-of-Presence Based on Context*

As discussed in Sect. 3.2.1, Varshavsky *et al.* proposed a PoP scheme called AMIGO in which two peer devices first establish a security association using Diffie-Hellman key agreement and then verify their co-location by comparing the RSSI values of data packets observed by both using a WiFi access point which both devices can connect to. If the observed RSSI values are similar enough, the peers are determined to be co-located. The security of the scheme is based on the observation that fluctuations in the RSSI values are spatially limited and can therefore be used to verify co-presence of devices in each other's proximity. Later work, however, has shown [82] that as RSSI is an aggregate value over several individual measurements, it can be predictable or even influenced by a remote adversary [63]. A practical limitation of the AMIGO approach is also that peers need to be located relatively close (less than one metre) to each other in order for the verification to work. This limits the applicability of their approach as peer devices in many cases are located in the same room but farther away from one another than the required distance for pairing.

The co-location verification method by Narayanan *et al.* [100] uses the notion of *location tags*, i. e., information items obtained by peers from their ambient context. As mentioned in Sect. 3.2.1, they discuss diverse possible context modalities for constructing location tags, but analyse only location tags based on the header information of WiFi-packets observed on a WiFi access point commonly observed by the peers seeking to verify co-location. This limits the applicability of

their approach to such scenarios in which both peers can have access to the same access point in vicinity. The practicality of their scheme is also dependent on how much traffic the WiFi access point used generates, as low-traffic WiFi APs may not generate a sufficient amount of packets for establishing proofs of presence in a timely manner.

3.8.3 *Distance-Bounding Approaches*

The notion of *distance bounding* was introduced by Hu *et al.* as a defence against wormhole attacks in mobile ad-hoc networks [55]. Distance bounding utilises the fact that the speed of light is limited and uses highly accurate timing measurements of message round trip times to establish an upper limit on the distance of peers involved in the distance bounding protocol. This requires from the peers the ability to make such high-accuracy measurements and is usually not possible without specialised hardware that is suitable for this purpose, limiting its usability on regular smartphones that do not have such hardware support.

3.9 SUMMARY AND CONCLUSION

In this chapter we discussed Proofs-of-Presence (PoPs) based on context for co-location verification in the setting of On-line Social Networks (OSNs) and Location-Based Services (LBSs). In Sect. 3.1.2 we showed that commonly used context modalities for context-based PoPs are vulnerable to so-called *context guessing attacks*. In these attacks an adversary utilises information it has gathered from a target context ahead of time to construct fabricated PoPs to fool the verifier to accept it as a proof of the adversary's presence in its proximity. We then discussed two countermeasures against such context guessing attacks: surprisal filtering and the use of longitudinal ambient context modalities for constructing more robust PoPs that are more difficult to successfully guess for the adversary.

Surprisal filtering is based on profiling the context(s) of the verifier and using this profiled information for estimating the probability with which the adversary could have successfully guessed the particular context values used in the PoP. Such PoPs that are deemed too easy to guess by the adversary can then be rejected to reduce the risk of accepting adversarial PoPs as genuine.

The use of longitudinal ambient context modalities like ambient luminosity and audio is complementary to surprisal filtering and serves to increase the entropy of context-based PoPs. In contrast to many previous works on context-based PoPs, their use is not bound to specific infrastructure and therefore widely usable also in a variety of different mobile use cases. As for longitudinal context-based PoPs the context is monitored for a modestly longer period of time and en-

tropy extraction is based on observable changes in the monitored context modalities, our approach is able to extract sufficient entropy from the context to provide for usable but robust context-based proofs of presence.

4

CONTEXT-BASED AUTHENTICATION OF IOT DEVICES

Currently deployed mechanisms for *device pairing*, i. e., the process of establishing an authenticated security association between devices typically rely on active involvement of the user. In these approaches, the user needs to either enter a code on the to-be-paired devices, compare authentication strings displayed by the devices, or, *demonstratively identify* [6] the devices by bringing them close to each other so that they can perform an exchange of security keys over a location-limited channel like NFC. The involvement of the user is required to eliminate man-in-the-middle attacks. Requiring the user to be involved in this process is, however, tedious and error-prone, as users easily make mistakes or do not pay sufficient attention to ensure a correct outcome of the authentication process (e. g., by not checking that displayed authentication codes actually match, but just 'clicking through' authentication prompts displayed by devices). Especially when the amount of devices that need to be paired grows, traditional approaches requiring user involvement in pairing become quickly impractical. What would therefore be desirable is a device pairing approach requiring no explicit interaction from the user.

The importance of reliable and user-friendly device pairing methods is at the same time gaining in importance since the amount of individual devices that need to be connected is constantly growing due to the emergence of the so-called Internet of Things (IoT). More and more manufacturers are bringing new kinds of devices on the market with which users can monitor and control many aspects of their 'smart homes'. Examples of IoT devices include, e. g., smart power plugs, smart light bulbs, motion sensors, window/door sensors, smart thermostats, weather monitoring stations, smart coffee makers and many more household appliances with added network connectivity. Another important device group that is gaining in importance are *wearables*, i. e., devices like fitness trackers, smart watches and other similar devices that are worn by users. IoT devices and wearables are likely to play a significant role in the computing infrastructures of smart homes or small offices in the future and therefore having secure but usable methods requiring no explicit interaction by the user for pairing them with the user's trust domain are increasingly important. This is particularly important as many IoT devices and wearables do not have traditional user interfaces or even displays for facilitating traditional pairing approaches.

The goal of this work is to devise a context-based authentication approach for pairing IoT devices and wearables that does not require explicit interaction of the user to function. Our authentication approach is based on profiling the ambient context of devices that are located in the same physical space like a room by performing context measurements from which *context fingerprints* are extracted. Since these fingerprints are likely to be similar if the devices are co-located, they can be used together with appropriate error-correcting codes to establish a shared authentication secret with which the involved devices can authenticate the presence of the pairing counterpart in the same physical environment and thereby confirm membership in the same *trust domain* like, e. g., the set of the user's IoT devices.

4.1 BACKGROUND

Approaches for device pairing that do not require explicit user interaction can be divided into three broad categories: *key pre-sharing*-based approaches, *demonstrative authentication via proximity* and *implicit context-based pairing*. In the following, we will review most prominent approaches belonging to these categories.

4.1.1 *Key Pre-Sharing*

Key pre-sharing based approaches (e. g., [35, 23, 76, 139]) are based on distributing key material on devices before their deployment in the field. These mechanisms target mainly devices acting as nodes in WSNs. The basic key agreement mechanism introduced by Eschenauer and Gligor [35] is based on randomly assigning a number of keys from a common key pool to individual devices before their deployment in the field. The objective of their scheme is to allow deployed nodes to autonomously form secure link keys, as it is in most deployment scenarios impossible (or at least very inconvenient) for the user to be actively involved in the pairing of individual devices.

The scheme relies on the birthday paradox, due to which adjacent nodes share with high likelihood common keys and can subsequently use these to establish link keys. The scheme by Eschenauer and Gligor has been subsequently extended by various aspects improving WSN key agreement [23, 76, 139].

Key pre-sharing-based approaches for key agreement have, however, limitations which prohibit their use in IoT scenarios for practical reasons. A major obstacle it that establishing sufficiently large key pools covering all device vendors would be a daunting task due to the ever-growing number and heterogeneity of device manufacturers. Different vendors have vastly varying practices regarding security design, implementation and operational security. It is unlikely that a unified trust framework covering all manufacturers that would

be needed to control and administer the common key pool could realistically be established.

However, even if single IoT device vendors were to decide to establish a joint or vendor-specific key pool for pre-sharing-based key agreement for devices they produce, this would still fail to provide an adequate solution for separating devices according to their respective trust domains. As it is likely that neighbouring trust domains (like the smart homes of two users living in neighbouring apartments with overlapping wireless range of their smart devices) may contain devices from the same device manufacturer, key pre-sharing based approaches could not be used to provide separation between devices having pre-shared keys drawn from a joint key pool but belonging to different trust domains.

4.1.2 *Demonstrative Authentication*

An initial paper by Stajano [135] introduced the so-called *resurrecting duckling* security model that discusses establishing device-to-device security associations between a 'controller device' and a newly purchased device that is introduced to the user's trust domain. Stajano's paper predates the era of smartphones, so in more recent set-ups a smartphone or a tablet would typically assume the role of this 'controller device'.

To establish device-to-device security associations, the model introduces the notion of *imprinting*. This refers to a process in which the unprovisioned device is 'imprinted' on the controller device by performing a key exchange and establishing a shared secret between them. The original scheme proposed to use a physical 'touch' with direct electrical contact between the devices for performing this initial key agreement constituting the imprinting step.

A similar approach for establishing security associations has been adopted by recent pairing schemes that require the user to *demonstratively authenticate* [6] devices that shall establish a security association. This is achieved by requiring devices to be placed very near to one another for pairing to succeed. A number of approaches for realizing such demonstrative authentication have been proposed.

One approach utilizes communication interfaces like NFC that have a very limited communication range, allowing peer devices to exchange keys in plaintext over the NFC channel. The NFC channel provides a certain level of protection against eavesdropping attackers as the used radio technology has a very short range and is therefore challenging to observe by external adversaries.

Other proposed approaches utilize contextual features, i. e., properties of the ambient environment like WiFi or Bluetooth beacons, luminosity or audio that can be sensed with devices' sensors. A scheme proposed by Varshavsky *et al.* [143] uses observed fluctuations in the

RSSI values of WiFi access points within wireless range for authenticating key agreement between co-located peer devices. The security of this scheme is based on the fact that these fluctuations are correlated only between devices that are located in close proximity of one another, thereby allowing only such devices to pair successfully that are located next to each other.

In subsequent work it became clear, however, that since RSSI values are a highly aggregate measure over an entire frequency band, they may be predictable or even influenced by a remote (but within wireless range) adversary who has information about the positioning of the devices involved in the pairing [63]. The security of RSSI value-based schemes remains therefore questionable. This is why Mathur *et al.* [82] improved the scheme by focusing on random fluctuations in the RF field of, e. g., frequency modulation (FM) radio or television broadcasts for extracting random bits for the authentication secret. These fluctuations are, due to the physical properties of the RF field, correlated only within a distance corresponding to half of the wavelength of the used RF signal and can therefore not be predicted by adversaries located farther away. In practice this distance is relatively short (ca. 15-35 cm for FM radio and television broadcasts), which limits the applicability of this approach in typical IoT scenarios, as IoT devices are usually placed farther away from each other.

4.1.3 *Context-Based Pairing Approaches*

In contrast to demonstrative authentication, context-based authentication approaches do not require devices to be placed close to one another for pairing to succeed. These approaches utilize the contextual information that devices can sense from from their ambient environment utilizing their on-board context sensors. By comparing these measurements, devices can verify that they are are co-located. The underlying assumption is that devices that can observe the same context belong to the same trust domain, e. g., the set of IoT devices in the smart home of the device owner. The main advantage of context-based pairing is given by its usability, as it provides a way to establish security associations between devices without the need for explicit user involvement. For establishing pairings between devices it is sufficient that the devices are placed in the same physical perimeter, e. g., in the same room. This provides a convenient way to separate devices belonging to different trust domains, as the context measurements of devices not located in the same room will be too different from one another, so that context-based authentication will not succeed. We will analyse the impact of contextual separation on the security of context-based pairing schemes in detail in Sect. 4.6.

4.1.3.1 *Context-Based Zero-Interaction-Authentication*

As discussed in Sect. 3.2.2, Truong *et al.* [140] and Shrestha *et al.* [131] have introduced schemes for device co-location verification in zero-interaction authentication (ZIA) scenarios. In such settings, peer devices already have established a security association and the main target of co-location verification is to prevent *relay attacks* against the ZIA protocol. These approaches use contextual modalities like luminosity, audio and other directly observable properties of the context for verifying co-presence of devices in the same context during session authentication.

The setting in typical IoT on-boarding use cases is, however, different. There devices typically do not have a pre-existing trust relationship and therefore need to establish an initial security association with other devices in the same trust domain. The schemes of Truong *et al.* and Shrestha *et al.* therefore aren't directly applicable to IoT device pairing, as they assume the peers to be mutually trusted. In IoT scenarios we can't assume devices initially to be trusted, but must first use context-based authentication to establish the authenticity in terms of trust domain membership of the involved devices.

4.1.3.2 *Context-Based Key Agreement*

The first practical scheme for key agreement utilizing contextual information was presented by Schürmann and Sigg [124]. They used measurements of the surrounding audio environment to allow two co-located devices A and B to establish a shared secret. In their scheme, information about sound energy level changes in different frequency bands in the surrounding audio environment is encoded into a *context fingerprint* f which is then used as a secret for hiding a randomly selected secret key a into a *fuzzy vault* [64]. They utilize a code-offset construction in which peer A selects a secret a and first encodes it as a codeword c using a Reed-Solomon code, i. e.,

$$a \xrightarrow{Encode} c \tag{4.1}$$

The resulting codeword c is then subtracted from the context fingerprint f of the peer A to obtain an *opening value* δ that allows peer device B to retrieve the secret.

$$\delta = f \ominus c \tag{4.2}$$

A then transmits δ to B. Using it and its own fingerprint f', the other peer B will be able to retrieve the secret by subtracting δ from f' and decoding the resulting codeword c' to recover a, i. e.

$$c' = f' \ominus \delta \tag{4.3}$$

$$c' \xrightarrow{Decode} a' \tag{4.4}$$

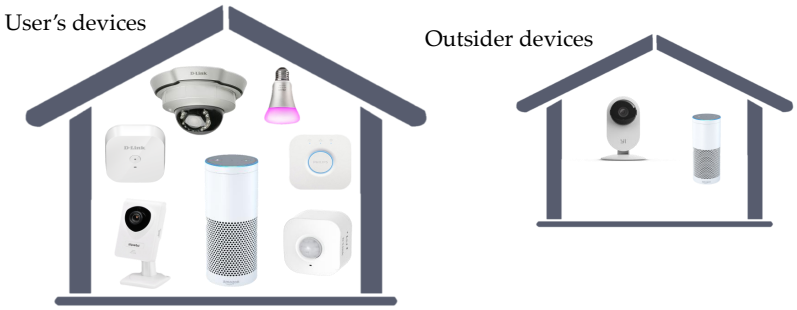


Figure 4.1: IoT device pairing scenario

Due to the error-correcting capability of the used Reed-Solomon code, $a' = a$ only if the Hamming distance $dist(f, f')$ of the fingerprints f and f' is less than the error-correcting capability t of the used Reed-Solomon code, i. e.,

$$a = a' \iff dist(f, f') \leq t \tag{4.5}$$

Using the fuzzy vault, the Schürmann and Sigg scheme allows two devices that are able to make sufficiently similar audio observations to agree on a common secret. Other devices that are not in the same context will not be able to retrieve the secret, as their fingerprints will not be sufficiently similar to be able to successfully retrieve the secret from the fuzzy vault.

4.1.4 *Problem Description*

In this work we focus on the problem of how to use contextual information observed by devices in the same physical environment like a room to establish a secure pairing between the devices. We consider two possible application scenarios: pairing of stationary IoT devices and pairing of wearables.

4.1.5 *IoT Device Pairing Scenario*

The typical IoT device pairing scenario is shown in Fig. 4.1. The user's IoT devices are located in the user's home, which is enclosed by walls and windows, and form a *trust domain*. The devices use their onboard contextual sensors to make measurements about their ambient context in order to authenticate their mutual presence in the same environment and establish a security association between them. The pairing must be, however, performed in a way that outsider devices outside the user's home (e. g., in a neighbouring apartment) are not able to establish a successful pairing with any of the devices in the user's trust domain.

4.1.6 *Wearable Device On-Boarding Scenario*

The second scenario we consider is related to the on-boarding of wearable devices like smart watches, fitness trackers, etc. that are typically constantly worn by the user for most of the time. The goal here is to allow a wearable device to use ambient context measurements to authenticate a security association with the smartphone of the user. In this scenario we assume that the user regularly carries his smartphone with him, so that the wearable device and the smartphone are co-located in the same context most of the time. The goal of the pairing process is that after a specific *gestation period*, the smartphone and wearable device can successfully verify that both devices belong to the same user and thus the same trust domain.

4.2 ADVERSARY MODEL

4.2.1 *Adversaries in IoT Pairing*

In the IoT device pairing scenario, adversary $\mathcal{A}$ is an IoT device within wireless range of the user's IoT devices, however, located *outside* the user's home, e. g., in a neighbouring apartment.

4.2.1.1 *Benign Adversaries*

In the benign case, $\mathcal{A}$ is an un-configured IoT device, merely trying to establish a pairing with other devices within wireless range. This class of adversaries have no malicious intent toward the user's trust domain, but should nevertheless not be allowed to establish authenticated security associations with devices in it in order to avoid, e. g., unintended disclosure of sensitive user information to outsiders.

4.2.1.2 *Malicious Adversaries*

Recently, a novel class of malware specifically targeting IoT devices has emerged, infecting IoT devices like IP cameras to form *IoT botnets*. For example, the *Mirai* IoT malware has been reported to have infected large numbers of IoT devices for using them as bots in a botnet for launching massive distributed denial of service (DDoS) attacks [3]. Adversary $\mathcal{A}$ can therefore be an IoT device that has been compromised with malware and intentionally attempts to pair with devices in the user's home domain. The adversary's target is to infiltrate the trust domain in order to expose sensitive information about the user or compromise devices in the user's trust domain and use them for nefarious purposes.

4.2.1.3 *Assumptions*

While we assume that $\mathcal{A}$ is not able to directly observe the context of the user's devices, there may be external sources like the outdoor lighting conditions that influence the observable context of both the user's devices and $\mathcal{A}$. We do therefore not assume that there would be a complete contextual separation between $\mathcal{A}$ and the user's devices.

We also assume that $\mathcal{A}$ does not have the capabilities to mount *targeted attacks* against the user's trust domain, i. e., using special equipment like directional high-fidelity microphones or telescopes to remotely measure the contextual parameters in the user's home, e. g., through a window. As we assume $\mathcal{A}$ to be a regular IoT device, it does not have the hardware and software capabilities for realizing such attacks.

It is possible that from time to time there are *visitors* to the user's smart home, carrying possible wearable devices with them. Therefore we have to assume that adversary $\mathcal{A}$ (both benign and malicious) might be intermittently present in the user's home for limited amounts of time. During this time, it will be able to observe the context in the same way as the user's IoT devices. However, the amount of time that $\mathcal{A}$ can spend in the user's home is limited and significantly less than what legitimate IoT devices spend in the user's context.

4.2.2 *Adversaries in Wearable Device On-Boarding*

4.2.2.1 *Benign Adversaries*

In this scenario, benign adversaries are wearable devices that have not been paired yet with a smartphone trying opportunistically to find a smartphone to pair with. They have no further motivation than to find a host smartphone for themselves.

4.2.2.2 *Malicious Adversaries*

In the wearable scenario, a malicious adversary $\mathcal{A}$ is an attacker trying to play a man-in-the-middle or impersonation attack on the user's smartphone and his wearable device in order to obtain sensitive information exchanged between the wearable device and the smartphone.

4.2.2.3 *Assumptions*

We assume that adversary $\mathcal{A}$ can be occasionally present in the same context as the user's smartphone or wearable, e. g., when the user and his devices are visiting the same place where $\mathcal{A}$ is located, and

can hence observe the same contextual parameters as the user's devices. However, $\mathcal{A}$ is not able to continuously follow the user, so that the amount of time that $\mathcal{A}$ spends co-located with the user's devices is limited and significantly smaller than the amount of time the user's smartphone and his wearables are co-located in the same context. During the time that $\mathcal{A}$ is not in the same context as the user's devices (i. e., not in the same room) it is not able to observe the same contextual parameters.

4.3 SYSTEM DESIGN

In this section we lay out the goals, requirements and design of our *context-based authentication approach* for IoT devices and wearables entitled ConXPair, which is based on utilising sensed observations from the shared context of two devices A and B to allow these devices to gradually authenticate their mutual membership in the same trust domain.

4.3.1 *Goals and Requirements*

As mentioned in the beginning of this chapter, there is a need for a device-to-device authentication approach which would allow devices belonging to the same trust domain to establish pairings *without the need for explicit interaction* from the user, as this provides clear usability benefits, especially in emerging IoT environments, in which there might be dozens if not hundreds of different IoT devices in the trust domain and setting up security associations manually would require significant effort. We envisage therefore a context-based authentication approach, which *does not require direct user intervention*.

Earlier context-based pairing solutions (e. g., [124]) are one-time authentication schemes, i. e., they are executed only once, based on context data aggregated during a relatively short period of time. The security of these schemes relies on the assumption that adversary $\mathcal{A}$ is not present in the same context and can not observe the same contextual parameters as the pairing peers A and B during the pairing process. This assumption, however, does not apply in general in the IoT and wearable scenarios referred to above in Sects. 4.1.5 and 4.1.6, as we assume adversary $\mathcal{A}$ to be intermittently present in the same context as devices A or B. Device A can therefore not verify the authenticity of device B instantaneously by merely verifying B's presence in the same context at a particular point in time. Instead, to verify membership of B in the same trust domain as A, it needs to verify the *sustained co-presence* of B in A's context.

4.3.2 *Solution Intuition*

To realize the verification of sustained co-presence, we propose an *iterative key evolution scheme* in which A and B repeatedly perform context-based authentications to verify co-presence during particular time periods. The idea behind this scheme is, that initially, devices establish *unauthenticated* security associations with *any* peers within wireless communication range, including devices in the same trust domain as well as possible adversarial devices. Each of these security associations is assigned an authenticity rating that is initially zero, indicating that the membership of the counterpart in the same trust domain has not been verified yet. Using the key evolution approach, the authenticity rating is incremented with each successful authentication iteration. Since devices A and B on average have a significantly higher probability of succeeding in the context-based authentication (by virtue of the fact that they spend significantly more time co-located in the same context than any adversary $\mathcal{A}$), over time the mutual authenticity ratings of A and B will diverge from that of adversary $\mathcal{A}$, so that the security association between A and B can be accepted as genuine and associations with $\mathcal{A}$ are rejected and discarded. In the following, we will discuss the details of our proposed key evolution approach.

4.3.3 *Context-Based Key Evolution*

The context-based key evolution is based on the assumption that two co-located peer devices A and B can utilize the common information contained in measurements of their ambient context to verify co-presence. By iteratively repeating the co-presence verification, A and B can with each successful iteration evolve their mutual pairing key and increase their belief in the authenticity of the peering counterpart. Here, authenticity refers to membership in the same trust domain.

The approach is based on *context fingerprints* w and w' that A and B, respectively, extract from their context measurements. Context fingerprints can be based on a number of contextual modalities like measurements of the ambient luminosity, audio, etc. We will discuss context fingerprints in detail in Sect. 4.4.

The context-based authentication approach consists of three distinct phases: initialisation, key evolution and key acceptance.

4.3.3.1 *Initialisation*

In the initialisation phase, devices A and B use Diffie-Hellman key exchange [29] to establish an *unauthenticated* shared secret K^0_{AB}. After establishing this shared secret the purpose of the key evolution is to gradually increment the belief in the authenticity of the counterpart

by repeatedly evolving the shared secret by performing context-based authentications.

4.3.3.2 *Key Evolution*

As mentioned above, A and B have established a shared key. This key at iteration i is denoted with K_{AB}^{i}. The goal of each iteration is to evolve key K_{AB}^{i} to an evolved key K_{AB}^{i+1} utilizing context fingerprints w and w' extracted from the ambient context of devices A and B, respectively. Inherent differences in the way involved devices observe their ambient context and possible sensing errors cause their fingerprints to be similar but *not identical*. However, for evolving the key a unique shared secret is required.

To agree on a unique shared secret, A uses a *secure sketch* as introduced by Dodis *et al.* [31] to transfer *error-correcting information* P about its fingerprint w to B, which allows B to eliminate possible sensing errors causing deviations between the fingerprints. A secure sketch is a pair of algorithms $SS(\cdot)$ and $SRec(\cdot,\cdot)$ that are based on an error-correcting code (ECC). The secure sketching operation $SS(w)$ outputs error-correcting information P that can be used to reconstruct w together with a value that is sufficiently similar to w, i. e.,

$$SS(w) = P \tag{4.6}$$

For reconstructing the original value w, the operation $SRec$ is used. It is able to reconstruct w given the error-correcting information P and any value that is sufficiently similar to w, i. e., for which the Hamming distance to w is below the error-correcting capability t of the ECC, i. e.,

$$SRec(w', P) = w \iff dist(w, w') \leq t \tag{4.7}$$

After B has reconstructed the context fingerprint w of A the key evolution process proceeds by evolving the shared key $K_{AB}^{i} \rightarrow K_{AB}^{i+1}$. In our approach we utilise a password-based key exchange (PAKE) protocol in this step, using a key evolution key K' derived from K_{AB}^{i} and the fingerprint w as the shared secret in the key exchange. K' is derived from w using a keyed hash h with the shared key K_{AB}^{i} as the key:

$$K' = h_{K_{AB}^{i}}(w) \tag{4.8}$$

We have to employ a PAKE protocol for key evolution because the entropy of context fingerprints w used for authentication purposes may not be sufficient to resist off-line brute-force attacks. If we were, e. g., to use K' directly as the evolved key, i. e., if we were to set $K_{AB}^{i+1} = K'$, an adversary $\mathcal{A}$ masquerading as legitimate peer device B could potentially retrieve K' even without knowledge of w simply by performing a brute-force attack by enumerating likely values of w,

deriving corresponding key candidates and testing whether the key candidate will decrypt subsequent messages correctly. Since PAKE protocols are resilient against brute-forcing attacks, we use PAKE to derive the evolved key K_{AB}^{i+1} as detailed below. Concretely, we take use of the *Encrypted Key Exchange (EKE)* protocol using exponential key exchange of Bellovin and Merrit [10] to perform the key exchange for obtaining the evolved key. One iteration of the key evolution approach is shown in Fig. 4.2.

1. Both A and B monitor their contexts and extract context fingerprints w and w' from their respective measurements.

2. A utilises the secure sketching operation $P = SS(w)$ to extract error-correcting information P from its fingerprint and sends this to B (message ①).

3. B uses its fingerprint w' and the error-correcting information P in the operation $w^* = SRec(w', P)$ to eliminate deviations between its fingerprint w' and A's fingerprint w to obtain a reconstructed fingerprint w^*. If A's and B's fingerprints are sufficiently similar, i. e., if $dist(w, w') \leq t$, the reconstructed fingerprint w^* will be identical to A's original fingerprint, i. e., $w^* = w$, otherwise not.

4. To determine whether the context authentication was successful, and, if so, to evolve the shared key, A and B derive *key evolution keys* K'_A and K'_B, respectively, using a keyed hash function, as indicated in (4.8), i. e.

$$K'_A = h_{K_{AB}^i}(w) \qquad K'_B = h_{K_{AB}^i}(w^*) \tag{4.9}$$

   If the original fingerprints of A and B were sufficiently similar $w = w^*$, and consequently also the key evolution keys will be identical.

5. Devices A and B will then both separately generate random exponents R_A and R_B, respectively, and calculate corresponding residuals $(\alpha^{R_A} \mod \beta)$ and $(\alpha^{R_B} \mod \beta)$, where α and β denote the public generator and modulus of the scheme.

6. Devices A and B will then encrypt their respective residuals as $E_{K'_A}(\alpha^{R_A} \mod \beta)$ and $E_{K'_B}(\alpha^{R_B} \mod \beta)$ using their corresponding key evolution keys K'_A and K'_B.

7. A sends then a message (②) to B containing its identifier A, and the encrypted residual.

8. Device B calculates a candidate key K_B^+ as

$$K_B^+ = \alpha^{R_A R_B} \mod \beta \tag{4.10}$$

9. B sends to A (message ③) its encrypted residual and a random challenge n_B encrypted with the candidate key K_B^+.

10. A decrypts B's encrypted residual with its key evolution key K'_A as

$$(\alpha^{R_B} \mod \beta) = E^{-1}_{K'_A}\left(E_{K'_B}(\alpha^{R_B} \mod \beta)\right) \tag{4.11}$$

and calculates a candidate key K_A^+ as

$$K_A^+ = \alpha^{R_A R_B} \mod \beta \tag{4.12}$$

11. Device A then retrieves B's challenge by decrypting it with its candidate key K_A^+as

$$n_B = E^{-1}_{K_A^+}\left(E_{K_B^+}(n_B)\right) \tag{4.13}$$

and sends a message (④) containing a random challenge n_A and B's challenge n_B encrypted with A's candidate key K_A^+.

12. Device B decrypts message ④ with its candidate key K_B^+ as

$$n_A, n_B = E^{-1}_{K'_B}\left(E_{K_A^+}(n_A, n_B)\right) \tag{4.14}$$

and verifies that the challenge n_B decrypts correctly.

13. B then encrypts A's challenge with its candidate key K_B^+ and sends it to A (message ⑤).

14. A decrypts B's reply to its challenge as

$$n_A = E^{-1}_{K_A^+}\left(E_{K_B^+}(n_A)\right) \tag{4.15}$$

and verifies that it decrypts correctly.

15. If at any point in the execution of the protocol any of the responses to the challenges can't be successfully verified, the verifying party will abort the protocol and notify the peer of the failure. Otherwise the protocol is successful and both devices will use their candidate keys as the new evolved shared key K_{AB}^{i+1}, i. e.

$$\text{Device}A: \; K_{AB}^{i+1} = K_A^+ \qquad \text{Device}B: \; K_{AB}^{i+1} = K_B^+ \tag{4.16}$$

4.3.3.3 *Key Acceptance*

With each iteration of the key evolution approach, the belief in the authenticity of the counterpart (i. e., belief that the counterpart is sustainedly present in the same context and thus member of the same trust domain) is increased. To determine how many evolution iterations are required for attaining a trusted pairing is dependent on the desired authentication strength and the probability of success of the adversary $\mathcal{A}$.

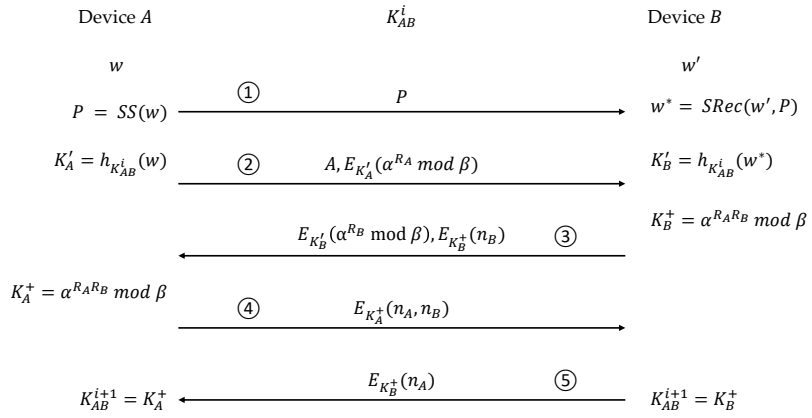


Figure 4.2: Context-Based Key Evolution

REQUIRED AUTHENTICATION STRENGTH The required strength of the authentication is a security parameter that determines how difficult it should be for adversary $\mathcal{A}$ to (falsely) authenticate with peers in the user's trust domain. This requirement should be determined based on the use case at hand. In our scenarios, we take an industry-standard solution like Bluetooth pairing with a six-digit authentication PIN as the baseline, as this is a widely used and accepted set-up for device-to-device pairing solutions. In this pairing approach the adversary has at most a one-in-a-million chance of succeeding in pairing by random guessing of the authentication PIN. This is roughly equal to an entropy of 20 bits from the viewpoint of $\mathcal{A}$. We will adopt this requirement also for our solution, so that the adversary $\mathcal{A}$ shall have a probability P_{max} of at most 2^{-20} for falsely succeeding in the context-based authentication process.

ADVERSARIAL SUCCESS PROBABILITY As we will discuss in more detail in Sect. 4.5, adversary $\mathcal{A}$ will in most settings have a non-negligible average probability P_{succ} in succeeding in a single iteration of the context authentication. The value of P_{succ} is dependent on the contextual setting of each particular deployment environment and has to be estimated empirically using conservative estimates before deployment of the context-based authentication scheme in practice.

Using the above parameters we can now formulate a condition for the minimum amount of required authentication iterations for obtaining a secure pairing. If the success probability P_{succ} of adversary $\mathcal{A}$ is higher than the required authentication strength P_{max}, the context-based key evolution has to be repeated at least k times so that the aggregate success probability $(P_{acc})^k$ of $\mathcal{A}$ falls below P_{max}, i. e.,

$$k = \arg\min_i \left((P_{acc})^i \right) \leq P_{max} \tag{4.17}$$

Here, we can assume $\mathcal{A}$'s success probability to be the product of individual iterations' probabilities, as the probabilities of individual iteration trials are independent from each other.

INTERMITTENTLY CO-LOCATED ADVERSARIES While k denotes the minimum required number of successful authentication iterations for adversaries *outside* the context in which A and B are located, an additional requirement needs to be taken into account, if it is possible that adversary $\mathcal{A}$ may pay occasional visits to the target context, or, if in the wearable device scenario, the user (and consequently, his wearable devices) may spend time in a context in which $\mathcal{A}$ is present. The success probability for fingerprints that $\mathcal{A}$ generates during such periods is significantly higher than P_{succ}, approximately equal to the success probability of the user's legitimate devices. We need therefore to discount any fingerprints that $\mathcal{A}$ generates during such periods of co-location with the user by increasing the minimum required number of successful authentication iterations by l iterations, where l denotes the maximum number of fingerprints that $\mathcal{A}$ can be expected to observe during its visits to the user's context. The parameter l therefore depends on the assumed longest aggregated duration of $\mathcal{A}$'s visits as well as the duration of sampling a distinct fingerprint and their distribution over time.

4.4 CONTEXT FINGERPRINTING

Our context fingerprinting approach is designed for two prominent contextual modalities applicable for context-based pairing: ambient audio and luminosity. The fingerprinting scheme is inspired by Schürmann and Sigg [124], but differs from it in several aspects. Whereas the scheme in [124] requires tight time synchronization, our scheme does not, due to the more longitudinal approach taken by our fingerprinting scheme. Similarly, the Schürmann and Sigg scheme is intended to extract a significant amount of entropy within a short time period in order to be used as a cryptographic key, whereas our fingerprints use their longitudinal orientation to cover the aspect of *sustained presence* over extended periods of time. The resulting fingerprints will thus typically capture effects in the respective modalities that originate from the user's actions in the target context (e. g., switching lights on and off, talking, walking, etc.). User actions are inherently random events and therefore difficult to predict by adversaries, even for advanced attackers utilising profiled information about users.

The longitudinal nature of our fingerprinting and key evolution approach has also the advantage that it provides the possibility take also adversaries into account that are occasionally co-located with the user's devices, i. e., visiting adversaries. This is a clear benefit over

earlier approaches, where pairing is performed during a short period of time and the security of the pairing relies on the assumption that the adversary is not present in the context of the user's devices at the time of pairing [143, 124]. This might be a difficult assumption to fulfil, especially in scenarios related to wearable devices, where one can't always assume devices to be located in a protected private space like the home of the user. In contrast, the longitudinal approach can gracefully also handle situations in which the adversary pays occasional visits to the user's context.

4.4.1 *Context Measurements*

To derive longitudinal context fingerprints, devices A and B continuously measure their context, yielding a sequence of context measurements $m_1, m_2, \ldots$ of the monitored context modality like, e. g., ambient luminosity or sound energy level. Each measurement m_i is associated with a timestamp, denoted $t(m_i)$.

4.4.2 *Context Quantisation*

To derive the context fingerprint w, the context measurements can be quantised in a number of different ways. In this work, we have experimented with two distinct approaches, encoding of level changes, as well as peak-based list encoding. The advantage of the former is that it provides a steady stream of fingerprint bits. However, the relation of '1' and '0' bits in the resulting fingerprint using this approach is imbalanced, leading to a lower entropy per fingerprint bit from the viewpoint of the adversary. The peak detection approach can provide fingerprints with a balanced distribution of '1' and '0' bits, but may take longer to generate fingerprint bits, as explained below.

4.4.2.1 *Level-Change Encoding*

In this context quantisation approach, fingerprint bits are generated based on significant changes in the average value of the monitored context modality. For this, the sequence of context measurements $m_1, m_2, \ldots$ is divided into windows of length d. For each window, the average context parameter value c_i is calculated as the average of the measurements falling inside the window:

$$c_i = \frac{\sum_{j=id+1}^{(i+1)d} m_j}{d} \tag{4.18}$$

Based on the sequence of context parameter averages $c_0, c_1, \ldots$, a corresponding fingerprint bit sequence $b_1, b_2, \ldots$ is generated as follows. A fingerprint bit b_i is set to '1' if there is a significant change in the value of value c_i in comparison to its predecessor c_{i-1}, i. e., if

their relative difference is larger than threshold Δ_{rel} and the absolute difference larger than threshold Δ_{abs}, i. e.,

$$b_i = \begin{cases} 1, & \left|\frac{c_i}{c_{i-1}} - 1\right| \geq \Delta_{rel} \wedge |c_i - c_{i-1}| \geq \Delta_{abs}, \\ 0, & otherwise. \end{cases} \tag{4.19}$$

The resulting fingerprint bit sequence is then divided into discrete fingerprints w_i, i. e., subsequences of length l bits.

4.4.2.2 *Peak-Based List Encoding*

An alternative context quantisation approach is a variation of the *list encoding* scheme proposed by Mathur *et al.* [82]. In this scheme, device A first detects significant peaks in its monitored context parameter values and then uses these to encode a *randomly chosen* fingerprint w by encoding it with the help of the time of time offsets $t(m_i)$ of measurements m_i representing peaks in the parameter value. In the Mathur *et al.* scheme, '1' bits are are encoded with peaks representing local maxima, while 'o' bits are represented by downward peaks representing local minima in the monitored context modality (they used the amplitude of the temporal channel variations of RF channels as the context modality). However, since context modalities like audio often do not contain clear local minima, we modify their scheme slightly. To encode 'o' bits we randomly pick a roughly equal amount of non-peak observations from the observed measurements and use these to encode zero bits. The encoded fingerprint consists therefore of the time offsets of measurements corresponding to peaks (local maxima) representing one bits and to non-peaks representing zero bits.

For the context-based key evolution scheme (cf. Fig. 4.2) this means that device A will send in addition to the error-correcting information P also the time offsets of the encoded fingerprint to B, who will then also first perform peak detection on its own context measurements and use the time offsets to decode its fingerprint w'. Offsets corresponding to identified peaks in B's measurements will be encoded as '1' bits and as 'o' bits otherwise.

The rationale for these context quantisation approaches is the following. Two devices present in the same context for a longer period of time will likely also observe changes in context parameters in a similar way. If the user, e. g., switches on the lights in the room where the devices are located, the increased in luminosity will be sensed by all devices located inside the room, whereas devices outside the room will not be able to observe it. Fingerprint bits generated this way will therefore be similar only between co-located devices. This reasoning applies also to fingerprints based on audio. Alternating sound levels caused by chatter, silence and other ambient sounds will lead to fingerprint bits being generated in a pattern that is similar between

devices in the same audio context (e.g., the same room), whereas outsider devices will not be able to observe nor replicate these patterns.

Similar reasoning is applicable also to wearable devices and smartphones that are usually carried together. The context in which the devices are simultaneously located changes constantly as the user moves, but these changes will be sensed in parallel in a similar way by all devices being worn by the user.

4.5 EVALUATION

To analyse the feasibility of the presented context-based authentication approach we performed a number of experiments in various settings to investigate the similarity of context fingerprints extracted from ambient luminosity and audio in real-world contextual settings.

4.5.1 *Evaluation Metrics*

We evaluated the context fingerprinting approaches with regard to factors that affect the security and practicality of our context-based authentication approach, namely fingerprint similarity and entropy of obtained fingerprints.

4.5.1.1 *Fingerprint Similarity*

Fingerprint similarity plays a role for our scheme in two aspects: for one, the fingerprints of co-located user devices A and B need to be sufficiently similar that a secure sketch with error-correcting capability of t bits can be used to eliminate differences between the observed context fingerprints of A and B so that the context-based authentication can succeed. This means that due to (4.7) the Hamming distance of A's and B's fingerprints needs to be less than t. On the other hand, we need also to verify that the similarity of the fingerprints of either A or B is sufficiently different from adversary A's fingerprint so that it can't succeed in context-based authentication with either of the user's devices.

4.5.1.2 *Fingerprint Entropy*

As we will show in Sect. 4.6, the entropy of the context fingerprints affects the resilience of the authentication result against guessing attacks by the adversary. The used fingerprint needs to have sufficient entropy to ensure the security of the authentication result. On the other hand, also the *entropy rate*, i. e., the number of entropy bits per fingerprint bit plays a role for the practicality of the authentication scheme, as higher entropy rates allow to aggregate sufficient entropy faster, so that shorter fingerprints can be used and consequently less time is required for performing individual authentication iterations.

4.5.2 *Experiment Set-Up*

In our study we used smartphones running the Android operating system equipped with dedicated context data collection software as surrogates for IoT and wearable devices. This approach was selected, since smartphones provided the necessary programmable facilities for implementing adjustable and flexible sensing of ambient context parameters and storing these data. The context data collection app on each device continuously measured the ambient luminosity and sound energy levels and sent the collected data regularly to a back-end server for off-line data analysis.

In these experiments the test devices were placed in different domestic and office environments to simulate IoT device pairing scenarios, whereas for the wearable device pairing scenarios test persons carried two different data collection devices with them during which contextual data were collected.

4.5.2.1 *IoT Pairing Scenario*

The target of our evaluation in the IoT device pairing scenario was to investigate whether IoT devices located in the same room can successfully observe sufficiently similar context fingerprints for successful context-based key evolution. We also wanted to verify that there is a sufficient difference in the similarity of fingerprints between co-located devices and adversaries so that only co-located peers will be able to successfully pair in the analysed settings.

The IoT pairing scenario was tested in two separate experiments utilising different approaches for context quantisation. In these experiments the relative placement of the devices and their position within the target rooms were varied and different context quantisation methods were tested for fingerprint extraction. Table 4.1 shows an overview of the placement of the used data collection devices in both experiments.

In the office setting, devices A and B were placed on the wall of an office room in three metres distance from each other. Devices $\mathcal{A}_i$ simulating adversaries were placed in nearby rooms having no direct visibility to the room of the target office. In the home setting, devices A and B were located in the living room of the test user's home. The devices $\mathcal{A}_i$ simulating adversaries in a neighbouring apartment were placed in another room of the house, either on a different floor or separated by a light-weight door from the living room.

In Experiment 1, also adversary devices $\mathcal{A}_1$ were placed on the window of the target room, with the luminosity sensor facing towards the outside in order to collect luminosity measurements about the outdoor luminosity affecting also the ambient lighting in the target room. To have consistency on how external sources may affect indoor lighting conditions, such rooms were selected that had large windows

Table 4.1: Set-up of IoT scenario experiments

EXP.	QUANTISATION	SETTING	DEVICE	PLACEMENT
1	level-change	Office	Device A	User's office
			Device B	User's office
			$\mathcal{A}_1$	Outdoor light
			$\mathcal{A}_2$	Adjacent office
			$\mathcal{A}_3$	Coffee room
1	level-change	Home	Device A	Living room
			Device B	Living room
			$\mathcal{A}_1$	Outdoor light
			$\mathcal{A}_2$	Studio, 2nd floor
2	peak-based	Office	Device A	User's office
			Device B	User's office
			Device C[1]	Adjacent room 1
			Device D[1]	Adjacent room 1
			Device E[1]	Adjacent room 1
			$\mathcal{A}_1$	Adjacent room 2
2	peak-based	Home	Device A_1	Living room
			Device B_1	Living room
			Device C_1	Living room
			Device A_2	Kitchen
			Device B_2	Kitchen
			$\mathcal{A}_1$	Storage room

[1] Devices C, D and E were used in the evaluation also as adversaries $\mathcal{A}_2 \ldots \mathcal{A}_4$ for devices A and B, and vice versa.

facing the same direction, allowing outdoor lighting to illuminate all rooms used in the experiment in a similar way during daytime hours.

4.5.2.2 *Wearable Device Scenario*

For this scenario we simulated the ambient contextual environment of typical wearable devices. Test users were equipped with smartphones simulating wearables that users carry with them or wear on their body. Two distinct settings were considered: a 'smart watch' scenario, where device one device simulates a smart watch, and the other device takes the role of a regular smartphone, and a 'cycling' scenario to simulate use cases related to wearable fitness trackers.

In the 'smart watch' scenario users were equipped with two smartphones which they carried with them continuously. Device A took the role of a smart watch that is worn on the user's wrist. It was placed in a translucent carrying pouch so to allow it to be constantly exposed to the ambient light in the same way a wrist-worn device would be. Device B on the other hand, was used like a regular smartphone.

In the 'cycling' scenario the smartphones were used to simulate fitness trackers, a highly popular class of wearable devices. There are currently dozens of different vendors offering wearable fitness tracking products for measuring physical characteristics like heart rate, steps taken, activity hours etc. In our scenario, we consider a cyclist using fitness tracking devices to record and monitor his physical performance during his workout. One smartphone was attached on the side of the test user's bicycle helmet with the light sensor facing outwards, while the other device playing the role of, e. g., a heart rate sensor, was placed in a translucent carrying pouch on the chest of the cyclist, also facing outward. In the cycling scenario, ambient light and audio measurements were made during the workouts of the cyclist typically covering a distance of approximately 15 kilometres and lasting for roughly one hour at a time.

4.5.3 *Datasets*

In experiment 1 of the IoT device pairing scenario luminosity and audio measurements were collected during several weeks, sampling the context once every 1 s. From the collected data we extracted context fingerprints using the level-change encoding approach (cf. Sect. 4.4.2.1) and a time window of $d = 120\,\mathrm{s}$ (cf. (4.18)) for generating fingerprint bits. We then compared the average bit differences of the resulting fingerprints between co-located devices A and B as well as between A and B and the adversary devices $\mathcal{A}_i$.

Experiment 2 focused on the peak-based list encoding approach for context quantisation (cf. Sect. 4.4.2.2) and therefore utilised only measurements in the audio modality. This is because luminosity

changes in the context typically occur gradually over time, making identification of distinct peaks in the luminosity measurements very challenging. In this experiment, the ambient sound energy level was recorded every 100 ms to allow for accurate detection of peaks. The data collection encompassed in total 12 different devices over a period of 30 days, resulting in more than 8000 hours of context measurements.

In the wearable device scenario, data traces were collected from co-located devices carried by test persons in various dynamic and static contexts, e. g., while walking, or commuting with public transport, as well as during stays in the home and office contexts. In the 'cycling' scenario, 10 traces of back-and-forth journeys on a fixed route of approximately 15 kilometres were recorded. The traces covered a period of several weeks, with varying road and weather conditions. Since the context changes in the wearable scenarios typically much faster than in more static scenarios, we applied a shorter time window of $d = 5\,\mathrm{s}$ for the luminosity data and a slightly longer time window of $d = 6\,\mathrm{s}$ for audio, resulting in fingerprints of 665 to 784 bits per exercise trace for luminosity and 501 to 551 bits for audio data.

4.5.4 *Fingerprint Similarity*

In the first part of our evaluation we examined the fingerprint similarities between co-located and adversarial devices in our experimental settings.

4.5.4.1 *IoT Device Pairing Scenario*

As we will show in Sect. 4.5.5, night time context measurements contain too little contextual events to be useful in generating fingerprints with sufficient entropy, as user activity and changes in illumination are very scarce. Night time data are therefore not well suited to be used for context-based authentication. We focus our analysis therefore on the most active times in the day, equalling roughly to the business hours between 8 a.m. and 6 p.m. in the office setting and 6 a.m. and 10 p.m. in the home setting. The fingerprint similarities during those times in experiment 1 are shown in Tab. 4.2.

In the office setting, co-located devices clearly have more similar fingerprints in comparison to adversaries $\mathcal{A}_i$, being on average 95.0 % for luminosity-based and 91.8 % for audio-based fingerprints. However, for luminosity-based fingerprints the similarity of adversary device $\mathcal{A}_2$ located in the adjacent office is also relatively high, 88.7 %. This is likely due to the fact that light conditions outside affect the rooms' illumination in the same way and the impact of daylight typically dominates during business hours. In view of this fact it is interesting that adversary $\mathcal{A}_1$ observing the outdoor light nevertheless has a lower fingerprint similarity than $\mathcal{A}_2$. This is explained by the

Table 4.2: Average fingerprint similarity between the co-located and adversary devices in the IoT scenario experiment 1

Office setting, 8 a.m. to 6 p.m.	LUMINOSITY	AUDIO
A and B	95.0 %	91.8 %
$\mathcal{A}_1$	70.0 %	-
$\mathcal{A}_2$	88.7 %	71.7 %
$\mathcal{A}_3$	68.3 %	62.6 %
Home setting, 6 a.m. to 10 p.m.	**LUMINOSITY**	**AUDIO**
A and B	82.9 %	87.5 %
$\mathcal{A}_1$	70.8 %	-
$\mathcal{A}_2$	70.6 %	77.0 %

placement of the adversary device on the outside window where its light sensor has a very wide angle of view on outside lighting conditions but won't observe the effects that building-related factors like the shadows it casts cause on indoor luminosity.

For audio-based fingerprints the difference between the similarity of co-located devices and adversaries $\mathcal{A}_i$ is clearer. Adversary $\mathcal{A}_2$ located in the adjacent office of devices A and B reaches an average fingerprint similarity of merely 71.7 % in comparison to 91.8 % for the co-located devices. This difference exists even though both rooms connect to a common hallway to which the doors were often kept open so that the acoustic isolation of the rooms was not perfect. The impact of acoustic isolation can also be seen from the average fingerprint similarity of adversary device $\mathcal{A}_3$ that was located in a coffee room that was farther away from the target room, so that it was acoustically better isolated from the co-located devices. The similarity of its fingerprints with the co-located devices is significantly lower, i. e., 62.6 %.

In the home setting, the fingerprint similarities between co-located devices were comparable, on the average 82.9 % for luminosity-based and 87.5 % for audio-based fingerprints. This was clearly better than for the fingerprints of the adversary devices $\mathcal{A}_1$ and $\mathcal{A}_2$, which could achieve at best bit similarity values of merely 70.8 % for luminosity-based and 77.0 % for audio-based fingerprints.

The results for experiment 2 were similar, however, limited to the audio modality. There, the similarity of co-located devices was on average 93.2 % in the home setting and 95.2 % in the office setting. The fingerprint similarity of adversary devices was in both settings lower, 86.1 % in the home setting and 67.9 % in the office setting. As with adversary $\mathcal{A}_2$ in experiment 1, the cause for the higher similarity of the adversary in the home setting is caused by insufficient acoustic

isolation of the adversary from the target room. This was caused by practical constraints of the experiment set-up, due to which the storage room in which adversarial device $\mathcal{A}_1$ was placed was separated from the other devices only by a light-weight door that had to be opened from time to time. However, as we will discuss in detail in Sect. 4.6, this allowed us to analyse the impact of contextual separation on the security of the context-based authentication.

In summary we can state that in all examined contextual settings there is a clear difference in the average bit similarities of co-located and adversarial devices. In all scenarios it would therefore be possible to use a secure sketch based on an ECC with an error-correcting capability corresponding to ca. 10 % to 13 % of the used fingerprint length. This would allow co-located devices according to (4.7) to succeed in context-based authentication, while adversary devices would fail to do so.

4.5.4.2 *Wearable Device Scenario*

In the 'wearable' scenario the observed fingerprint similarity of co-located devices was on average 92.6 % (ranging from 87.3 % to 96.7 %). This would mean that an ECC with error-correcting capability corresponding to ca. 10 % of the fingerprint length could be used to realise context-based authentication for devices in the wearable scenario.

In the 'cycling' scenario the similarity of fingerprints of co-located devices was on average 68.6 % (ranging from 62.8 % to 74.5 %) for luminosity-based fingerprints while for audio-based fingerprints it was 65.9 % (ranging from 63.6 % to 67.1 %). This suggests that in the set-up of the 'cycling' scenario, movements of the test person's body may introduce a significant amount audio-visual artefacts that are potentially sensed differently by devices attached on different parts of the body. This manifests itself in lower fingerprint similarity between the wearable devices, requiring the use of an ECC with a higher error-correction capability for context-based authentication to succeed.

As discussed in Sect. 4.2.2.3, we assume that in the wearable scenario the adversary $\mathcal{A}$ is occasionally present in the same context as the user's wearable devices. During this time it is able to observe the same contextual features and can therefore generate fingerprints that are equally similar to the user devices' fingerprints. The adversary therefore has an equal chance of succeeding in context-based authentication than legitimate devices. However, we also assume that the adversary is not able to constantly follow the user and therefore the time that $\mathcal{A}$ is able to observe the user's context is limited. To discount for these periods, the minimum number of required successful context-based authentication iterations k according to (4.17) needs to be increased by l which corresponds to the maximum amount of fingerprints an adversary $\mathcal{A}$ is expected to be able to generate based on data generated during these periods, as discussed in Sect. 4.3.3.3.

4.5.5 *Fingerprint Entropy*

In this work we denote with fingerprint entropy the amount of uncertainty that adversary $\mathcal{A}$ has about the fingerprints of co-located devices A or B. In this sense, fingerprint entropy is a measure of how difficult it is for the adversary to correctly guess a fingerprint $w_{\mathcal{A}}$ that is *sufficiently similar* to the user's device's fingerprint w (or w') to allow it to successfully authenticate with it.

Fingerprint extraction is dependent on observed contextual activity. Therefore the amount of entropy bits that can be obtained from the context typically varies depending on the hour of day. During periods with a lot of activity in the context, more entropy can be extracted, whereas during more inactive times like during the night, only little contextual entropy is available. Since the used context quantisation methods generate fingerprints in a different way, we shall analyse the entropy of the resulting fingerprints separately.

4.5.5.1 *Fingerprint Surprisal*

The level-change encoding-based context quantisation approach (cf. Sect. 4.4.2.1) generates fingerprint bits at a steady rate, as, according to (4.18), context measurements m_i are averaged to context parameter values c_i over windows of d measurements and for each such window a fingerprint bit is generated. However, as according to (4.19) the fingerprint bit values are dependent on the presence of significant changes in the monitored contextual parameter values c_i, the distribution of '1' and '0' bits is highly dependent on the amount of contextual activity during the measurement period. Therefore, e. g., during night time, as the context typically is silent and without user-related activity, '0' bits dominate in the generated fingerprints, leading to a very skewed distribution between '0' and '1' bits.

We have to assume that adversary $\mathcal{A}$ has knowledge of the typical distribution of '0' and '1' bits during particular times of day and need to take this into account when estimating the entropy of fingerprints. We do this by evaluating the *surprisal*, i. e., the self-information of distinct fingerprint bit values during a particular time of day. Surprisal is calculated by evaluating the occurrence probability of a particular bit value b in the fingerprint during a specific time of day, using a frequentist interpretation of probability. The probability of a particular bit value (i. e., '1' or '0') equals to the fraction of that bit's occurrences in context fingerprints during that time of the day. The surprisal of individual bit values is therefore defined as follows.

Definition 16 (Surprisal of fingerprint bits) *If B is a random variable modelling the occurrences of bit values in fingerprint w, then the* surprisal

$\sigma(b)$ associated with the occurrence of a fingerprint bit $b \in \{0,1\}$ is the self-information *of this value, i. e.*

$$\sigma(b) = I(b) = log\left(\frac{1}{P(B=b)}\right) = -log\left(P(B=b)\right), \tag{4.20}$$

and is measured in bits.

Definition 17 (Surprisal of a fingerprint)
The surprisal $\sigma(w)$ of a fingerprint w is the sum of the surprisal values of its individual bits, i.e.,

$$\sigma(w) = \sum_{b \in w} \sigma(b). \tag{4.21}$$

Whereas fingerprint surprisal refers to the unpredictability of a particular outcome, i. e., a concrete fingerprint instance, the entropy associated with a fingerprint of particular length is defined as the expected value of the surprisal a fingerprint of that length during the particular time of day. Formally, if W_l is a random variable over the values of possible fingerprints of length l bits during a particular time of day, then we can define the entropy of an l-bit fingerprint as follows.

Definition 18 (Entropy of fingerprints) *If W_l is a random variable modelling the possible values of fingerprints w of length $|w| = l$, we define the entropy $\mathbf{H}(W_l)$ of an l-bit fingerprint as the expected value of the surprisal of fingerprints of that size, i. e.*

$$\mathbf{H}(W_l) = \mathbb{E}(\sigma(W_l)) \tag{4.22}$$

To exemplify this, Fig. 4.3 shows the relation of 1-bits in the fingerprint and the associated fingerprint entropy of fingerprints of 60 bits with a window length $d = 120\,\mathrm{s}$ during different times of the day in the Office setting of experiment 1. As can be seen, the average fingerprint entropy varies significantly during the day. During night time, there are hardly any changes in the observed context measurements, resulting in only very few '1' bits being generated during this time of the day. Consequently also the entropy of fingerprints during night time is very low, only a few bits. During the day when there is significant activity in the context, numerous '1' bits are generated, resulting in a more balanced distribution of '0' and '1' bits and consequently also significantly more (> 50) entropy bits, corresponding to an entropy rate of ca. 0.9 bits or more of entropy per fingerprint bit.

For fingerprints extracted using the peak-based quantisation approach, the surprisal of fingerprint bits is constant, as encoded fingerprint bits (cf. Sect 4.4.2.2) are randomly selected and the distribution of '1' and '0' bits is roughly equal, leading to an average surprisal of

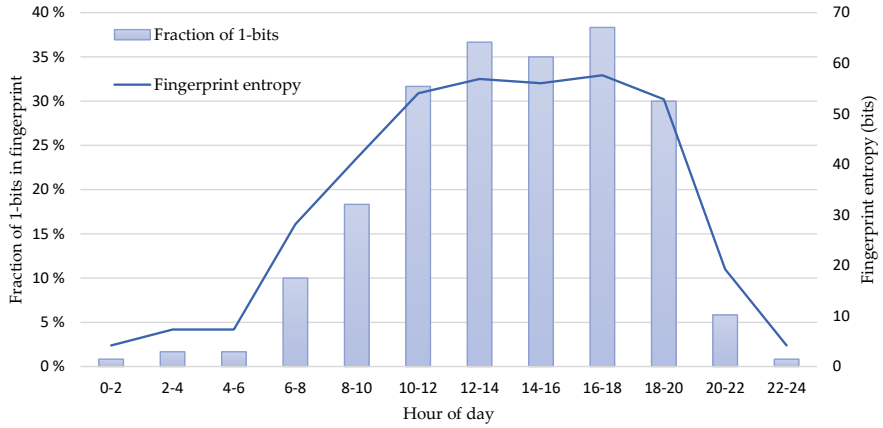


Figure 4.3: Fraction of 1-bits in and entropy of 60-bit fingerprints with window size $d = 120\,\mathrm{s}$ in the office setting of experiment 1 during different times of day

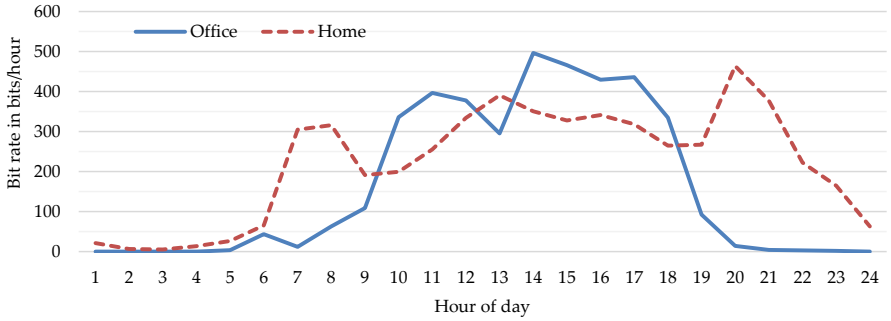


Figure 4.4: Bitrate of fingerprint extraction during different times of day in experiment 2 utilising peak-based list encoding

one bit per fingerprint bit. However, even though the level of contextual activity does not directly impact the surprisal of the fingerprint bits, it does have an effect on the amount of fingerprint bits generated per time unit, in contrast to the level-change-based quantisation approach in which the fingerprint bit generation pace is static.

The effect of the differences in contextual activity during different times of day on the rate at which fingerprint bits can be extracted from the context is shown in Fig 4.4. It shows the average rate of fingerprint bits extracted from the audio observations obtained during experiment 2 in dependence of the time of day. As can be seen, the more contextual activity there is in the context, the more bits are generated. This means that during the night when there is hardly any activity, only very few bits are generated. Significant amounts of bits are generated between 6 a.m. and 9 p.m. in the home setting and between 9 a.m. and 6 p.m. (on workdays) in the office environment. The average bit rates during these periods of time were 309 bits per hour in the home setting and 368 bits per hour in the office setting.

4.5.5.2 *Impact of Partial Information*

The above discussion about surprisal of fingerprints applies to adversaries that have knowledge of the typical distribution of fingerprint bits during the course of a day but do *not* have access to actual context measurements from the vicinity of the user's context in which devices A and B are located. However, as discussed in Sect. 4.5.4.1, environmental factors like, e. g., changes in outdoor lighting may affect the generated fingerprints in both the user's context and the contexts of nearby adversaries. Therefore the fingerprint $w_{\mathcal{A}}$ of an adversary located in a nearby room has in many cases in practice partial information about the fingerprint w of the user's device, which is reflected in a higher mutual similarity between these fingerprints.

The effective entropy of the fingerprint w can therefore be estimated by looking at the probability that a bit in adversary $\mathcal{A}$'s fingerprint's $w_{\mathcal{A}}$ is equal to the corresponding bit in the user's fingerprint w. As can be seen from Tab. 4.2, there is a 68.3 % to 88.7 % probability that a bit of a luminosity-based adversarial fingerprint $w_{\mathcal{A}}$ is equal to the corresponding bit in user's fingerprint w. This means that the average entropy rate of such fingerprints is 0.17 to 0.55 bits per fingerprint bit in the office setting from the adversary's point of view. In the home setting, the adversary's fingerprint similarity for luminosity-based fingerprints is 70.6 % to 70.8 %, equalling to an entropy rate of ca. 0.50 bits per fingerprint bit. For audio-based fingerprints, the corresponding similarity values are 62.6 % to 71.7 % in the office setting and 77.0 % in the home setting. These values correspond to an entropy rate of 0.48 to 0.68 bits per fingerprint bit in the office setting and 0.38 bits per fingerprint bit in the home setting. We will discuss in Sect. 4.6 how these entropy rates affect the security and practicality of the context-based authentication scheme.

In experiment 2, the results were similar, as the average fingerprint similarity of adversary $\mathcal{A}$ was 67.1 % in the office setting and 86.1 % in the home setting during the active times of a day. This corresponds to an average entropy rate of 0.32 bits per fingerprint bit in the office and 0.24 bits per fingerprint bit in the home setting.

4.6 SECURITY ANALYSIS

As discussed in Sect. 4.1, a number of schemes like *ProxiMate* of Mathur *et al.* [82] or the audio-based scheme of Schürmann and Sigg [124] utilise error-correcting codes for deriving authentication secrets for verifying the proximity of peers engaged in the authentication protocol. However, none of these works provide a systematic quantitative analysis on the security of these schemes under realistic real-world conditions. In the following we will analyse which factors need to be taken into account when evaluating the security of context-based authentication schemes based on error-correcting codes. We

will use the secure sketch-based construction presented in Sect. 4.3.3 as a basis for our discussion.

4.6.1 *Entropy Analysis*

The strength of the context-based authentication, i. e., the probability that adversary $\mathcal{A}$ is able to successfully authenticate with device A is dependent on the entropy of A's fingerprints from the point of view of $\mathcal{A}$. It is measured in terms of the *min-entropy* $\widetilde{\mathbf{H}}_\infty(W|P)$, where W denotes the probability distribution of A's fingerprints w and P denotes the error-correcting information for fingerprint w published by A.

Min-entropy is a 'worst-case' measure, i. e., it quantifies the entropy associated with those values of $w \in W$ that are easiest to be guessed by $\mathcal{A}$. It is an appropriate measure for the strength of the authentication scheme, as it specifically considers those outcomes that are the most favourable for the adversary $\mathcal{A}$ and therefore represents the minimal level of security that the scheme can provide.

4.6.1.1 *Entropy Loss due to Information Reconciliation*

As discussed in Sect. 4.3.3.2, devices A and B utilise a secure sketch based on an error-correcting code (ECC) like a Golay or Reed-Solomon code to perform *information reconciliation* [14] to eliminate deviations between their fingerprints w and w'. In this process, device A derives according to (4.6) error-correcting information P that enables B to localise and 'correct' deviating bits between w and its own fingerprint w', if the fingerprints are sufficiently similar, i. e., their Hamming distance is less or equal to the error-correcting capability t of the used ECC (cf. (4.7)). However, in this process, the error-correcting information P will inevitably also leak some information about A's fingerprint w, reducing thereby the entropy of w as an authentication secret. The amount of information leakage is bounded by the number of error-correcting bits, i. e., the bit length of P. For an $[n, k, 2t-1]$ ECC, where n denotes the length, k the dimension and t the error-correcting capability, this equals to $(n-k)$ bits of entropy loss [31]. This means that the higher the error-correcting capability of the used ECC is, the higher also the entropy loss caused by information reconciliation will be.

In terms of entropy loss, Reed-Solomon (RS) codes [114] provide an optimal trade-off between error-correction capability and entropy loss, as the code will incur for each symbol of error-correction capability an entropy loss of two symbols. In practice, using an approach as employed, e. g., by Schürmann and Sigg [124], in which fingerprint bits are encoded with the help of symbols of the RS-code, a code with error-correction capability of t bits will incur $2t$ bits of entropy loss.

Table 4.3 shows the required error-correction levels and associated minimum min-entropy for fingerprints in the examined experimental settings. As can be seen, the average required error-correction level varies between 5 % to 18 % for luminosity-based and between 6 % to 13 % for audio-based fingerprints. Taking into consideration the entropy loss incurred by information reconciliation this means that to reach sufficient authentication strength $P_{max} = 2^{-20}$ (cf. Sect. 4.3.3.3), the fingerprints w used in the context-based authentication need to have sufficient min-entropy $\widetilde{\mathbf{H}}_\infty(w)$ that their leftover entropy taking the entropy loss due to information reconciliation into account is at least 20 bits.

Consider, e. g., luminosity-based fingerprints in the office setting of experiment 1. As the average similarity of the fingerprints of co-located devices is 95 %, an error-correcting code with at least 5 % of error-correction capability has to be used, resulting in an entropy loss of at least 10 % of the bit length of w. To compensate for this entropy loss the used fingerprint w therefore has to have a min-entropy of at least 22.2 bits to reach a leftover entropy of at least 20 bits after information reconciliation. As the average entropy rate of fingerprints in this setting is 0.17 bits per fingerprint bit (owing to the fact that the similarity of adversary $\mathcal{A}_2$ is as high as 88.7 %), the minimum bit length required for fingerprint w is 131 bits. In contrast, in the home setting of experiment 1, luminosity-based fingerprints need to be only 61 bits long as the entropy rate of fingerprints in this setting is much higher, 0.51 bits per fingerprint bit, even though an error-correcting code with a significantly larger error-correcting capability (18 %) needs to be used and the entropy loss and required min-entropy are therefore considerably larger.

We can see that both the required amount of error-correction as well as the entropy rate of the used fingerprints play an important role for the required minimum length of the used fingerprints and consequently the time required for extracting sufficiently strong fingerprints for context-based authentication. Table 4.4 shows the required duration for extracting sufficiently long fingerprints in the examined experimental settings. In experiment 1, the required fingerprint extraction time is relatively long, due to the use of a large window of 120 seconds to generate fingerprint bits, resulting in a static bit rate of 30 fingerprint bits per hour. Generating sufficiently strong fingerprints takes therefore 100 to 262 minutes in the different experimental settings.

For experiment 2, which used the peak-based quantisation method for fingerprint generation, the bit rate is much higher, as more than ten times more fingerprint bits are generated on average per hour. This results in both experimental settings to a significantly shorter duration, i. e., 12 to 20 minutes, for aggregating the required fingerprint bits.

Table 4.3: Error-correction levels and minimum required min-entropy $\widetilde{\mathbf{H}}_\infty(W)$ of used fingerprints w to obtain $\widetilde{\mathbf{H}}_\infty(W|P) \geq 20$ bits needed for an authentication strength of $P_{max} = 2^{-20}$ in different experimental settings. The last two colums show the average entropy rates of fingerprints and the corresponding minimum bit length $|w|$ of fingerprint w to reach P_{max}.

EXP.	ERROR CORRECTION	$\widetilde{\mathbf{H}}_\infty(W)$	ENTROPY RATE	FINGERPRINT LENGTH $\lvert w \rvert$
LUMINOSITY				
Exp. 1 Home	18 %	31.3	0.50	61
Exp. 1 Office	5 %	22.2	0.17	131
AUDIO				
Exp. 1 Home	13 %	27.0	0.38	71
Exp. 1 Office	8 %	23.8	0.48	50
Exp. 2 Home	8 %	23.8	0.24	100
Exp. 2 Office	6 %	22.7	0.32	72

Table 4.4: Required time to extract required minimum-length fingerprints during active times of the day

EXP.	REQUIRED FINGERPRINT LENGTH	AVG. HOURLY BIT RATE	REQUIRED DURATION (min)
LUMINOSITY			
Exp. 1 Home	61	30	122
Exp. 1 Office	131	30	262
AUDIO			
Exp. 1 Home	71	30	142
Exp. 1 Office	50	30	100
Exp. 2 Home	100	309	20
Exp. 2 Office	72	368	12

4.6.1.2 *Privacy Amplification*

In our scheme, as discussed in 4.3.3, we assume that devices A and B establish an unauthenticated security association, e. g., using Diffie-Hellman key agreement, obtaining a shared secret key. However, as this key is *unauthenticated*, the purpose of the context-based key evolution process is to establish the authenticity of the pairing counterpart.

In some other approaches like, e. g., the work by Schürmann and Sigg [124], the context fingerprint w is used to exchange a secret key between the pairing devices A and B. The secrecy of the key is therefore dependent on the entropy of the used fingerprint w. This implies that the leftover entropy $\widetilde{\mathbf{H}}_\infty(W|P)$ of fingerprints after information reconciliation needs to be sufficient to resist off-line known-plaintext attacks, e. g., 128 bits.

However, as the error-correcting information P leaks information about w, the adversary will obtain *partial information* about the exchanged secret key. To obtain a cryptographic key over which the adversary *does not have even partial information*, so-called *privacy amplification* has to be performed. This can be done, e. g., by applying a universal hash function $h(\cdot)$ to obtain a close-to uniformly distributed shared secret over which $\mathcal{A}$ does not have even partial information. According to the Leftover Hash Lemma (LHL) [7], this privacy amplification step will incur an additional entropy loss of $\log \epsilon^{-1}$ bits, where ϵ is a security parameter determining how indistinguishable the distribution of the resulting secret keys is from the uniform distribution.

4.6.2 *Authentication Performance*

To evaluate the performance of our context-based authentication approach we consider two measures reflecting the security and usability of the proposed scheme: the false accept rate (FAR) and the false reject rate (FRR). FAR measures the rate at which adversary $\mathcal{A}$'s fingerprints $w_\mathcal{A}$ will falsely lead to a successful context-based authentication with A. This will happen if the Hamming distance of $w_\mathcal{A}$ to A's fingerprint w is less than the used ECC's error-correcting capability t, i. e., if $dist(w, w_\mathcal{A}) \leq t$. For the security of the scheme FAR is the most important measure, as a low FAR implies a low probability for adversary $\mathcal{A}$ being incorrectly authenticated as co-located with A.

FRR is a measure for the usability of the scheme in practice. It measures the rate at which a benign co-located device B fails to successfully perform a context-based authentication with its peer A. This is the case if the Hamming distance of B's fingerprint w' to A's fingerprint w is larger than the error-correcting capability of the ECC, i. e., if $dist(w, w') > t$. A high FRR implies that benign co-located peers often will fail to successfully authenticate, decreasing the usability and usefulness of the approach. Therefore, in order to be successful

in terms of security and usability, the context-based authentication scheme must seek to minimize both FAR and FRR.

In Sects. 4.5.4 and 4.5.5, we analysed the average similarities of fingerprints of co-located and adversarial devices in the different experimental scenarios and their impact on the amount of required error-correction and fingerprint length. We considered in this analysis only the *average* case. However, our evaluation revealed another factor that has not been explicitly taken into account in earlier works utilising contextual data for co-location authentication [82, 124]. It turns out that also the inherent variation in the similarity of context fingerprints needs to be taken into account, as this plays a significant role for the FAR and FRR of the context-based authentication scheme, as we will show below.

As the similarity values between co-located and adversarial fingerprints fluctuate around the average value, it happens incidentally that the fingerprint $w_{\mathcal{A}}$ of adversary $\mathcal{A}$ is in some cases sufficiently similar to the fingerprint w of A, i. e., $dist(w, w_{\mathcal{A}}) \leq t$, thus allowing $\mathcal{A}$ to falsely succeed in the authentication with A.

There are two main factors impacting adversary $\mathcal{A}$'s success probability:

ERROR-CORRECTION LEVEL Using an ECC with a lower error-correcting capability t decreases the probability that $\mathcal{A}$'s fingerprint $w_{\mathcal{A}}$ is sufficiently similar with A's fingerprint w, therefore making it more difficult for $\mathcal{A}$ to succeed in authentication and and consequently a lower FAR. However, reducing t makes it also more difficult for benign co-located devices to succeed in context authentication and thereby increases FRR.

FINGERPRINT LENGTH $|w|$ Using longer fingerprints has the effect of averaging out short-term fluctuations in the similarities of the used fingerprints, thus reducing the occurrence frequency of fingerprints that incidentally have higher similarity between the adversary's fingerprint $w_{\mathcal{A}}$ and A's fingerprint w, consequently reducing FAR.

Figure 4.5 shows the impact of these factors on the FAR and FRR values for different error-correction levels (5 %, 8 %, 10 %, 12 % and 15 %) in the office and home settings of experiment 2. From Fig. 4.5a we can see that best performance for the scheme is achieved for fingerprints with bit length 512. At an error-correction level of 10 % this set-up achieves an FAR of 0.2 % with an FRR of 0.8 %, meaning that only ca. two authentication attempts out of one thousand will be successful for the adversary, while less than one authentication attempt by benign co-located devices in a hundred will be falsely rejected.

In the home setting, shown in Fig. 4.5b, the performance of the scheme is clearly worse, especially with regard to FAR. With the same parameters, i. e., for 512-bit fingerprints with a 10 % error-correction

level, the scheme achieves an FAR of as much as 33.5 % at an FRR of 9.2 %. The reason for this is given by the limitations of the experimental set-up, as discussed in Sect. 4.5.4.1, as the acoustic isolation of the adversarial device was for practical reasons not sufficient. This shows that insufficient contextual separation has a very significant impact on the security of the context-based authentication scheme, since even with extremely long fingerprint lengths of, e. g., 4096 bits, the scheme can achieve in this setting an FAR of only 16.7 % at an error-correction level of 10 %.

These results show that even under favourable conditions as in the office setting of experiment 2 (cf. Fig. 4.5a), the adversary nevertheless has a non-negligible success probability ($P_{succ} \approx 0.2\,\%$ even in the optimal case) emphasizing the necessity to use key evolution over several authentication iterations as discussed in Sect. 4.3.3.2. The number of required authentication iterations according to (4.17) for different error-correction levels and fingerprint lengths is shown in Fig. 4.6. From Fig. 4.6a we can see that at 10 % error-correction level in the office setting, at least $k > 2$ authentication iterations are necessary for all fingerprint lengths to ensure that adversary $\mathcal{A}$'s success probability falls below $P_{max} = 2^{-20}$, which is required to reach the desired authentication strength comparable to, e. g., six-digit PIN-based pairing in Bluetooth.

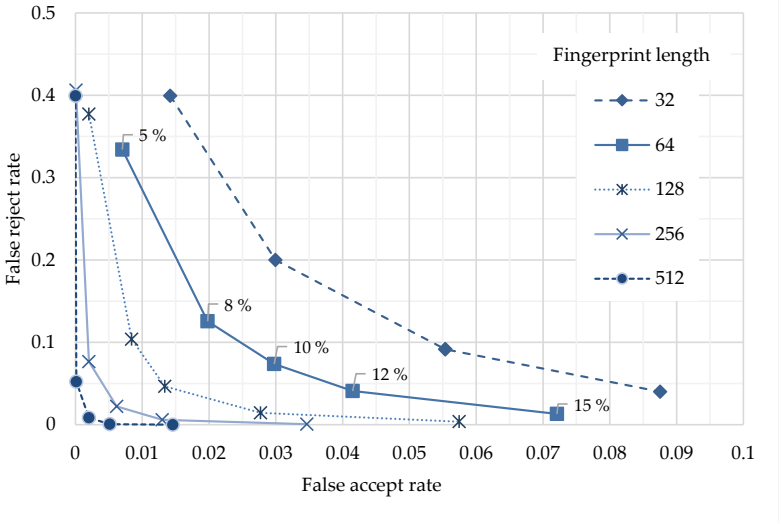
The effect of the insufficient contextual separation in the experimental set-up in the home setting is also visible in the amount of required authentication iterations, as shown in Fig. 4.6b. As can be seen, at the same 10 % error-correction level, 10 to 16 iterations would be necessary to reach a sufficient authentication strength for the pairing in this setting.

4.7 RELATED WORK

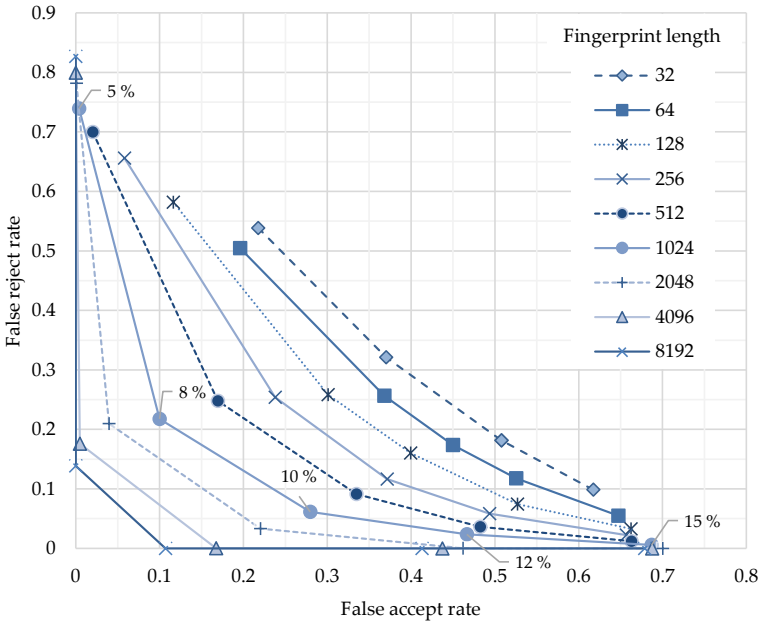
A number of schemes have been introduced for establishing security associations between devices in different scenarios. These can be roughly divided based on the way the key is established into key pre-sharing-based and context-based schemes.

4.7.1 *Key Pre-Sharing Schemes*

As mentioned in Sect. 4.1.1, Eschenauer and Gligor [35] were the first to introduce a scheme for key distribution in wireless sensor networks (WSNs) which was based on pre-distributing key material that was randomly sampled from a common key pool to sensor nodes before their deployment in the field. Owing to the *birthday paradox*, each sensor node would share with high likelihood one or more common keys with one or more neighbouring sensor nodes after their deploy-

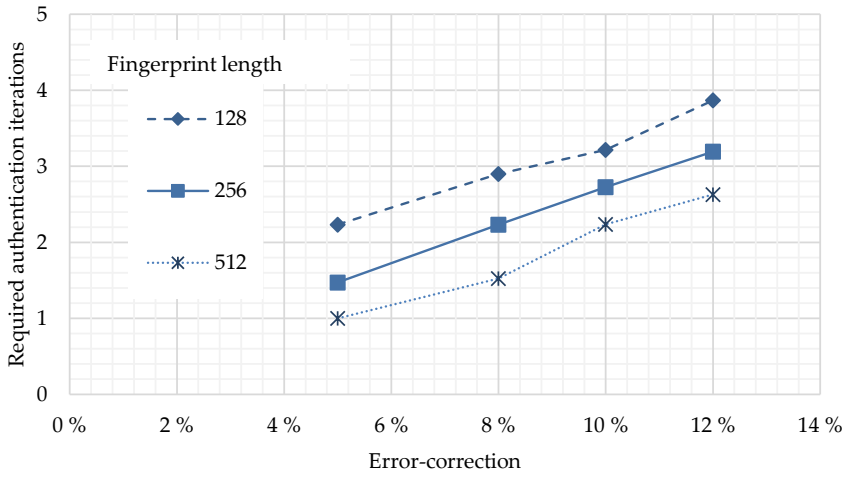


(a) Office

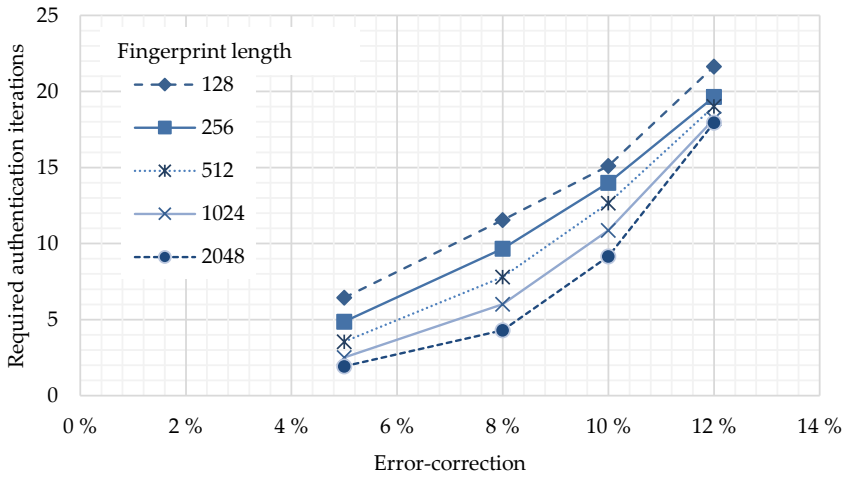


(b) Home (with insufficient contextual separation)

Figure 4.5: FAR vs. FRR for error-correction levels 5 %, 8 %, 10 %, 12 % and 15 % for different fingerprint lengths



(a) Office



(b) Home (with insufficient contextual separation)

Figure 4.6: Number of required authentication iterations to reach authentication strength corresponding to adversary success probability $P_{max} = 2^{-20}$

ment, allowing the nodes to use these keys in establishing a secure link key between them.

Chan *et al.* [23] extend the basic approach of Eschenauer and Gligor with three enhanced schemes, namely the *q-composite random key pre-distribution scheme*, the *multipath reinforcement scheme*, and, the *random pairwise key pre-distribution scheme*, providing improved resilience against various kinds of attacks and increasing the security of the key agreement, as well as providing robustness against the presence of compromised nodes in the network.

Traynor *et al.* [139] presented a variation of the scheme for settings in which one does not need to assume a fully homogeneous sensor node population, allowing *unbalanced* probabilistic key distribution. They also extended this approach to hybrid settings in which the approach can make use of special nodes acting as key distribution centres (KDCs).

Key pre-distribution-based schemes are mainly targeted at sensor nodes in WSNs, which are provisioned by a single organisation before their deployment in the field. Due to the reasons outlined in Sect. 4.1.1, these approaches are not well suited to scenarios involving IoT devices and wearables, as these typically originate from a very heterogeneous set of device manufacturers, lacking the organisational set-up and trust infrastructure required for key pre-distribution. The context-based authentication approach presented in this work does not have any such requirements regarding pre-distribution of keys and is therefore better suited for the targeted usage scenarios.

4.7.2 *Context-Based Schemes*

Varshavsky *et al.* [143] presented a scheme called *AMIGO* that uses the Received Signal Strength Indication (RSSI) of WiFi broadcast packets observed in the environment for allowing two devices to verify their proximity to each other. In this approach the peer devices measure and compare the signal strength fluctuations of observed WiFi packets to determine whether they are co-located or not and thereby authenticate a previously established secret key. The security of the scheme stems from the fact that variations in the observed RSSI values are correlated only if devices are located close to each other, thereby allowing to distinguish between devices that are located near to each other and other devices that are farther away.

The scheme was later extended by Kalamandeen *et al.* Their system, named *Ensemble* [65] utilises not only observations about incoming packets, but also active wireless transmissions of an ensemble of trusted wearable devices, increasing the reliability of proximity verification. However, subsequent works have shown that RSSI values are potentially vulnerable to manipulation of inference by a remote adversary [63], due to the highly aggregate nature of RSSI as a measure.

To overcome this vulnerability, Mathur *et al.* [82] therefore improved the approach in the *ProxiMate* system that uses the fluctuations in the RF-field of TV and FM radio transmissions for proximity verification. As discussed in Sect. 4.1.2, these approaches are only applicable for demonstrative identification scenarios in which the user has to place the devices close to each other (typically 15 cm to 35 cm) for the authentication to succeed. This limits, however, the applicability of these approaches in IoT device pairing scenarios in, e. g., Smart Home environments, due to the significant effort it imposes on the user to manually establish pairings between all of his potentially numerous IoT devices.

The work of Varshavsky *et al.*, Kalamandeen *et al.* and Mathur *et al.* was evaluated by Zenger *et al.* [148], who provide a thorough empirical evaluation and implementation of a proximity-based authentication scheme exploiting location-based channel randomness between an access point and two IoT devices, of which one is a trusted 'loaded' device used to authenticate the other 'unloaded' resource-constrained device. Their scheme improves security by removing dependence from external environmental sources that could potentially be influenced by the adversary.

Narayanan *et al.* [100] proposed another WiFi-based co-location verification approach, in which devices monitor WiFi broadcast packets and derive *location tags* from them. The peers will then compare these tags to determine whether they are co-located or not. This solution is, however, targeted at a scenario in which the goal of the participating devices is merely to establish evidence of co-location in a privacy-preserving manner. Consequently Narayanan *et al.* do not discuss pairing as such but assume the existence of appropriate pre-existing security associations between the participating peer devices.

Truong *et al.* and Shrestha *et al.* developed solutions for using rich context measurements for protecting zero-interaction authentication (ZIA) against relay attacks. Their approaches make use of contextual modalities to allow two devices to verify their presence in the same context. In this scenario, however, the involved devices are assumed to be trusted and have already established a security association. Their schemes are therefore not applicable in our scenario, where devices initially cannot make any assumptions about the trustworthiness or authenticity of the pairing counterpart.

Schürmann and Sigg [124] presented the first context-based authentication scheme that used context measurements in the key exchange process. Their scheme uses *context fingerprints* derived from measurements of the ambient sound energy level in several different frequency bands for exchanging a secret key between peer devices located in the same audio context. The scheme uses a *fuzzy vault* as introduced by Dodis *et al.* [64] that is based on a Reed-Solomon error-correcting code to allow devices with sufficiently similar con-

text fingerprints to successfully exchange a common secret key, while keeping adversary devices from doing so, as these will not be able to generate sufficiently similar context fingerprints.

However, Schürmann and Sigg did not provide a quantitative analysis of the security of their scheme. They do not consider that the secrecy of the exchanged secret key is dependent on the min-entropy of the used context fingerprints and that the adversary obtains partial information about the key due to the error-correcting information used in the key exchange. This increases the requirements towards the entropy of the fingerprint as well as makes it necessary to use privacy amplification to make sure the adversary doesn't have even partial information about the exchanged cryptographic key.

4.8 SUMMARY AND CONCLUSION

In this chapter we have discussed solutions that have been proposed for establishing security associations between devices. Traditional approaches rely on active involvement of the user, requiring the user to manually pair involved devices, e. g., by entering authentication codes or confirming them when displayed on the screen of a device. This process is in many cases tedious and error-prone, making it an unsuitable solution in the long run, as smart homes are expected to have more and more devices (dozens if not hundreds) that would need to be set up by the user.

Alternative solutions for key agreement like the ones proposed for WSNs [35, 23, 139] that use pre-shared keys drawn from a common random key pool, have practical limitations. It is very unlikely that IoT device vendors would be able or willing to set up a common key pool from which to sample required keys to be provisioned to devices before their deployment. On the other hand, these approaches do also not provide the possibility to separate trust domains of individual users in an effective and convenient way.

Other proposed approaches for device authentication seek to utilise contextual measurements like WiFi RSSI [143], RF field fluctuations [82] and audio [124] to establish the co-presence of the devices in the same context and use this as an authenticator of the fact that devices belong to the same trust domain. However, these works either have problems with regard to the security guarantees they provide [143], work only over a very short distance [82], or, do not adequately consider all factors that impact the security of the authentication scheme. Other schemes, on the other hand, address scenarios in which they already assume the existence of a security association between the involved devices [140, 131, 100].

To the best of our knowledge, this work is the first one to systematically analyse and quantify the factors that affect the security of context-based authentication schemes. Building and extending

on previous approaches, we present a context-based authentication framework that is based on context fingerprinting. The scheme utilises *secure sketches* that are based on error-correcting codes to allow peer devices present in the same context to evolve their unauthenticated pairing key and thereby gradually increase the confidence in the authenticity of the counterpart. In particular, our work takes into account the entropy loss introduced by the use of error correction to quantify the security of the authentication scheme in practice based on empirical evaluation data from real-world scenarios involving IoT device pairing and wearable devices.

The results highlight the importance of proper understanding of the properties of the context before deployment of the scheme in practice, as, e. g., the level of contextual separation between devices in the trust domain and outsiders plays a crucial role for the security of the scheme. However, by taking these factors into account, we show that using the iterative authentication approach it is possible to achieve reliable authentication and trust domain separation requiring no active involvement of the user in the process.

5

SECURITY MANAGEMENT IN IOT BASED ON DEVICE PROFILING

During the last years, the proliferation of the so-called Internet of Things (IoT) has been an emerging megatrend in the development of computing and communication systems. IoT encompasses a wide spectrum of different environments and contexts ranging from smart homes, buildings and smart city infrastructures to the industrial Internet. Recent forecasts predict that the number of connected IoT devices will globally grow to more than 20 billion devices by the year 2020 [45, 43].

Especially in the smart home setting IoT devices are enjoying growing popularity, as consumers install increasing numbers of internet-connected appliances in their homes in order to be able to monitor, control and automate aspects of their living environment. Examples of typical smart home IoT devices include smart power plugs, heating and air conditioning systems, security and surveillance systems, intelligent lighting, traditional kitchen appliances with added wireless connectivity and many other new and emerging device types.

Due to the increasing popularity of IoT devices, many new device manufacturers are entering the IoT market in order to benefit from the business opportunities provided by this rapidly growing market. Many of these entrants are new players who may not have significant previous experience in engineering products with internet connectivity and consequently lack the necessary expertise to apply good security designs and implementations in their products.

This has led to the situation that many devices being installed in user's homes have inherent security vulnerabilities. There is consequently an increasing amount of reports in the media about vulnerabilities in IoT devices that can be exploited by attackers (e. g., [25, 77]). Some software components are also widely reused in implementations of a wide variety of different types of IoT devices, so that a single flaw in the software implementation has the potential to be exploited in a very large number of different devices [127]. According to a recent report, many IoT devices are also using publicly-known private keys for authentication, rendering them susceptible to be compromised by adversaries taking advantage of this vulnerability [126]. Recently, also an entirely new class of malicious software specifically targeting IoT devices, so-called *IoT malware* has emerged. Some of the recent Internet-wide distributed denial of service (DDoS) attacks have been attributed to IoT malware like the infamous *Mirai*

botnet [51]. IoT malware typically utilise compromised IoT devices as bots in large distributed botnets for launching attacks.

All of the above developments have led to a situation in which an increasing number of IoT devices with exploitable security vulnerabilities are present in end-users' home networks, providing malicious adversaries entirely new opportunities to infiltrate and attack against users' networks and use the users' devices for other nefarious purposes. What are therefore needed are novel solutions for protecting users' local networks against threats posed by such vulnerable devices in order to stop devices from being compromised and proactively protect other devices in the network in case an adversary successfully manages to compromise a vulnerable device.

5.1 PROBLEM DESCRIPTION

The rapid development of the IoT device market has led to a situation in which new device manufacturers are bringing products to the market without having appropriate expertise about security designs nor skills for creating secure implementations. Many manufacturers are driven by a desire to bring their products quickly to the market in order to secure market share. This leaves often only little time for proper security designs and testing of products with security in mind. In addition, especially simpler devices are planned with a very limited budget, leaving virtually no resources for extensive security testing of the product. As a result, a significant number of products brought to the market have security vulnerabilities stemming from insecure design or flawed implementation at the time the devices are shipped.

5.1.1 *Insufficience of Software Patching*

The preferred way to deal with flawed implementations would be to fix them by applying appropriate security patches that eliminate software or design errors present in the device. However, in many cases device vendors do not provide such security patches in a timely manner. This can be either because they are unable to do so due to end-users not registering their products, so that manufacturers are unable to notify them about the availability of patches to their product. In other cases manufacturers may merely lack motivation for providing such updates in a timely manner, as this will not generate any immediate additional revenue to cover the costs of developing and distributing the patches.

Many device manufacturers also do not design their products with with software updates in mind, so that many products lack facilities for applying software updates automatically, leaving the responsibility for keeping their devices up-to-date with the end-users. Many

regular users, however, lack the necessary skills and motivation to make sure their devices are updated with the latest security patches.

5.1.2 *Need for Brownfield Solutions*

All of these factors lead to a situation in which there often are vulnerable IoT devices present in the users' home networks. Security solutions will need to take this into account and therefore be able to operate in situations in which one needs to assume that vulnerable devices with exploitable vulnerabilities are likely to be present in the network. Solutions need to enable such vulnerable devices to co-exist with other devices in the network during the whole lifetime of these devices. This setting mandates the security design to follow a *brownfield*[1] development approach, i. e., the developed mechanisms must be able to co-exist with devices that are inherently vulnerable to compromise as well as with other legacy devices and software components that users already have or are going to deploy in their home networks.

5.1.3 *Goal and Contributions*

We address the problem of vulnerable devices that are present in the user's home network through IoTSentinel, a system that is able to automatically identify the *device types* of devices that the user installs in his home network by fingerprinting the communication behaviour of each device and using machine-learning based classification models to match these fingerprints to known device types. After identifying the type of a device, IoTSentinel enforces mitigation measures for such devices that are known to have security vulnerabilities. This is done by applying appropriate traffic flow filtering in order to protect vulnerable devices from being compromised and preventing data leakage as well as protecting other devices in the network in cases where compromise of a vulnerable device cannot be prevented.

In this work we make following contributions:

- We present the design of IoTSentinel, a system for automatically identifying IoT devices when they are installed to the system and managing their security.

1 In software engineering, brownfield development refers to a setting in which a developed software system must co-exist in the immediate presence of legacy systems and therefore needs to take this into account in its architecture. The term is borrowed from civil engineering, where brown field land refers to a construction site where pre-existing buildings and infrastructure need to be taken into account when designing and erecting new buildings, in contrast to green field land, where no earlier structures exist.

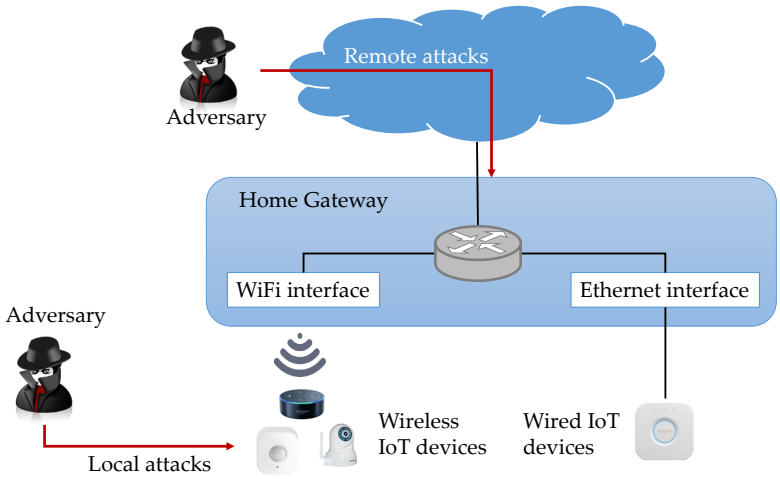


Figure 5.1: The IoTSentinel adversary model

- A framework for using traffic flow filtering to limit traffic flows of vulnerable IoT devices to protect them from compromise and protect other devices from potentially compromised devices.

We evaluate the performance of IoTSentinel based on a dataset of 27 real-world IoT devices and show its feasibility for effective security management of end-user home networks.

5.2 ADVERSARY MODEL

Our adversary model is shown in Fig. 5.1, showing a typical set-up in a local network in a home or a small office, in which IoT devices are connected to the home gateway router over WiFi or an Ethernet connection. We consider both local and remote adversaries.

A local adversary is located within the wireless range of the user's IoT devices or has possibly also intermittent physical access to them. A local adversary seeks to utilises vulnerabilities in the wireless interface of the IoT device to intrude and compromise the targeted device. Remote adversaries seek to attack against the user's IoT devices remotely over the network. While typical Small Office, Home Office (SOHO) routers do usually not allow external entities to directly contact devices in the internal network, it has been demonstrated that, e. g., malware on the smartphone of the user can be used to locate potentially vulnerable devices in the local network and use 'NAT hole punching' to allow the adversary to remotely connect to the vulnerable device for mounting attacks against it [134].

5.2.1 *Adversary Goals*

The goal of either local or remote adversaries is to compromise IoT devices in the user's network utilising exploitable security vulnerab-

ilities of devices. The target of the adversary is to exploit devices in order to either

- exfiltrate sensitive user data, security credentials or encryption keys,
- use compromised IoT devices for scanning for other vulnerable devices in the user's network,
- using compromised devices for launching attacks against other vulnerable devices in the local network or remote targets on the Internet, or,
- inject false or tampered information into the user's network in order to provoke desired reactions from the user.

5.2.2 *Assumptions*

We assume that IoT devices that the user installs in his network can have security vulnerabilities, but are initially uncompromised, i. e., benign. Therefore it will take some time until the adversary will locate a newly installed vulnerable device and identify and execute an appropriate exploit against the device. We therefore assume that the initial behaviour of the device when introduced to the user's network is benign for a sufficiently long period of time in order to allow IoTSentinel to collect sufficient genuine data about its communication behaviour to correctly identify the device's device type.

We also assume that the components of the IoTSentinel system are sufficiently well protected against targeted attacks against the IoTSentinel system itself. We therefore do not consider such attack scenarios in the context of this dissertation.

5.3 SYSTEM DESIGN

To protect the IoT devices in the local network of the user, IoTSentinel will perform following actions: it will use the communication behaviour of devices that are newly introduced into the system to fingerprint them and identify each device's *device type*. In this work, the notion of device type is defined to be the unique combination of a device's *make*, *model* and *software version*. Based on the identified type, it will then make a vulnerability assessment of the device, which is based on a repository of know vulnerabilities linked to particular device types. According to the result of the vulnerability assessment, IoTSentinel will enforce appropriate traffic filtering to protect devices from being infected and constrain potential damage in case devices do get infected by the adversary.

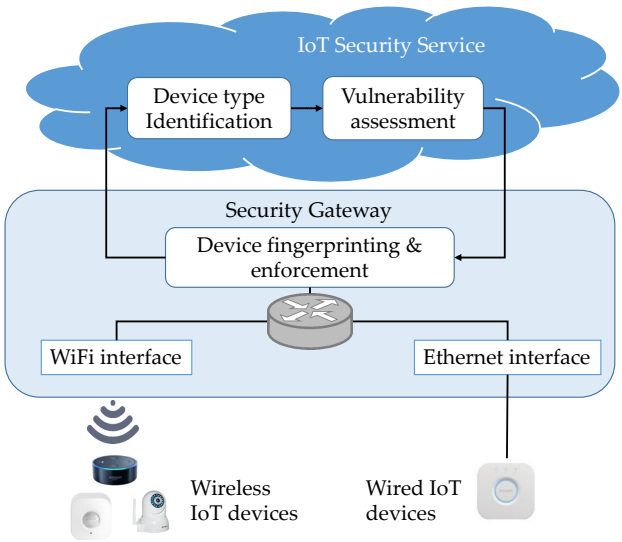


Figure 5.2: IoTSentinel system design

5.3.1 *System Components*

The system design of IoTSentinel is shown in Fig. 5.2. It consists of two main components: the Security Gateway (SGW) and the IoT Security Service (IoTSS).

5.3.1.1 *Security Gateway*

The Security Gateway takes the role of an ordinary SOHO WiFi router providing internet connectivity to all devices in the local network. The SGW can be realised in the form of a dedicated hardware device, replacing the WiFi router in the target environment, or, it could be installed to legacy routers in the form of a firmware update, if the hardware resources of the router are sufficient to support the additional functionality required by SGW. IoT devices in the user's home or office connect to SGW using WiFi or a wired Ethernet connection (For instance, many hub devices acting as gateways for devices like smart light bulbs using other wireless protocols than WiFi often are connected to the router over wired Ethernet).

The SGW monitors and fingerprints all devices in the local network connected to it. It sends the obtained *device fingerprints* to IoTSS who will identify each device's corresponding type and return it along with a security policy to be applied on the device type in question to SGW. It is the task of SGW then to enforce the security policy by applying appropriate traffic filtering on the traffic of IoT devices, as discussed in detail in Sect. 5.3.5.

The fingerprinting is based on monitoring the initial packets coming from an IoT device that is being installed in the user's network for a short period of time (e. g., 2 min) and deriving characteristic features that describe the communication behaviour of the device

during the set-up process. The rationale here is that the behaviour during this initial phase of device induction is relatively static allowing for reliable identification of each device's device type. As our evaluation in Sect. 5.4 shows, this communication behaviour is sufficiently characteristic for device types, allowing IoTSentinel to distinguish between them. The fingerprinting process is described in more detail in Sect. 5.3.2.

5.3.1.2 *IoT Security Service*

The IoT Security Service is a cloud-based server component operated by a specialised IoT Security Service Provider (IoTSSP). IoTSS aggregates fingerprinting information about known device types from all SGWs connected to it and performs device-type identification based on the device fingerprints that each SGW sends to it. It also maintains a repository of security policies for each device type that is known to have security vulnerabilities. Each policy determines an *isolation level* to be applied for the particular device type, along with possible fine-grained traffic filtering rules, e. g., denying access to particular ports on the device in question that are known to have exploitable vulnerabilities. We will discuss in more detail in Sect. 5.3.4 how the security policies are determined.

Based on the identified device type, IoTSS will retrieve the corresponding security policy from its repository and return the identifier of the device type along with the policy back to the SGW that requested the identification of the corresponding device fingerprint.

5.3.2 *Device Fingerprinting*

When IoT devices are first installed into a network, they typically undergo a set of configuration steps, during which they typically communicate with the device vendor's back-end system (e. g., to check for device firmware updates), or, perform service discovery to locate other devices and services in the local network. The pattern and semantics of these interactions are often vendor- and device-type-specific and can therefore be used to determine the type of each device. The goal of the fingerprinting approach is therefore to capture the characteristic patterns of these initial communications in a way that makes it possible to identify the type of each device.

The SGW acts as the WiFi router for the local devices and is therefore the endpoint of WPA2-encrypted connections over WiFi, allowing it to see all data frames sent over WiFi in plaintext. However, in many cases IoT devices use transport or application layer protocols like TLS or HTTPS to protect their communication with, e. g., the vendor's cloud service, so that actual packet payloads will often be encrypted. This mandates that the fingerprinting approach can util-

ise only packet header information that will be available in plaintext regardless of the presence of payload encryption.

Device fingerprinting is initiated when the SGW discovers that a device with a previously unknown MAC address is connecting to the local network. After this, SGW starts recording packets $p_1, p_2, \ldots$ originating from the device. From each packet p_i, a feature vector $(f_{1,i}, f_{2,i}, \ldots, f_{23,i})$ of 23 distinct features is extracted based on the header information of p_i. The features encode specific properties of the packet like used protocols, header flags, packet size as well as source and destination ports, as described in detail in [92]. The n first feature vectors are then combined to form device fingerprint $\mathbf{F}$.

After observing a device's communications for k seconds, the SGW will extract the fingerprint (if necessary, padding it with zeros, if less than n packets were observed) and send it to IoTSS for device-type identification.

5.3.3 *Device-Type Identification*

Device-type identification is performed by IoTSS using a two-step approach [92]: in the first phase, the device fingerprint $\mathbf{F}$ is tested against a set of device-type-specific classification models. Each of them is pre-trained by IoTSS using labelled fingerprints originating from the specific device type. Each classifier provides a binary prediction whether fingerprint $\mathbf{F}$ represents the classifier's device type or not. The device-type-specific classifiers are implemented using the Random Forest classification algorithm [16]. The testing of a device fingerprint against a Random Forest classifier is a relatively efficient operation, which allows fingerprint $\mathbf{F}$ to be quickly tested against a very large number of classifiers. The effectiveness of classification is important since also the number of distinct device types will likely grow to be very large.

Since it is possible that several classification models will provide a positive prediction, a second step is required, in which the edit distance of fingerprint $\mathbf{F}$ to a set of representative fingerprints from those device types for which the corresponding classifier provided a positive prediction [92] is compared. The device type with the lowest aggregate edit distance is then output as the predicted device type.

The two-step classification approach described above allows IoTSentinel to operate efficiently even if there are a large number of known device types. The first classification step acts as a quick prescreening phase for determining a potential subset of device types, thereby reducing the number of required edit distance calculations between the tested fingerprint and each device type's prototype fingerprints, which are significantly more time-consuming than the testing of a fingerprint against a Random Forest classifier [92].

5.3.4 *Vulnerability Assessment*

For each device type that is known to the IoTSentinel system the IoTSS will periodically perform vulnerability assessments to evaluate the potential threat level that is connected to individual device types. In performing the vulnerability assessment, IoTSS can utilise a number of complementary sources for vulnerability information.

5.3.4.1 *Vulnerability Repositories*

A straightforward way for identifying potential vulnerabilities of specific device types is to cross-check relevant records from publicly accessible vulnerability repositories like Common Vulnerabilities and Exposures (CVE) [95] or National Institute of Standards and Technology (NIST) National Vulnerability Database (NVD) [101] based on the device model information of each device type. Such vulnerability databases provide an automated and searchable way for the IoTSSP to find reported vulnerabilities for specific device types and evaluate their severity for deciding which mitigation measures to apply for particular device types.

5.3.4.2 *Penetration Testing*

In some cases, especially for particularly popular IoT device types, the IoTSSP may also employ automated or manual penetration testing in trying to identify security issues with popular device types. Due to the relatively high cost of this approach, however, penetration testing is not likely to be applicable to all device types in general. The advantage of this approach is that it provides relatively accurate information about found vulnerabilities, allowing an accurate assessment of related threats and required mitigation policies.

Penetration testing can be augmented with recent approaches for automated bug search in firmware binaries of devices [37], in cases where the firmware images of devices are available, e. g., from the device vendor's website.

5.3.4.3 *Vulnerability Crowdsourcing*

Another approach for aggregating vulnerability information about IoT devices is to crowdsource it from various sources like security advisories from device vendors, computer emergency response teams (CERTs), or, even security-oriented developer mailing lists like, e. g., *BugTraq* [19]. While crowdsourcing vulnerabilities from such sources provides a fast and up-to-date way of aggregating security information, it involves a high degree of manual work required to be done by security experts, therefore making it a very costly way of generating vulnerability assessments for device types. This would, however,

provide a business model for IoTSSPs as they could sell their expertise in aggregating vulnerability information for IoT devices, e. g., as a subscription-based service to customers, very much in the same way as anti-virus vendors sell subscriptions to their computer security products for workstations, desktops and laptops, where a core aspect of the products are regular updates to the malware signature databases required by these products.

5.3.5 *Enforcement*

The enforcement of device-type-specific security policies is realised by the SGW. Based on the security policy provided by the IoTSS, SGW generates software-defined networking (SDN)-based traffic flow rules that logically partitions the local networks into two virtual subnetworks, a *trusted* and an *untrusted* network. The traffic flow rules are enforced by a virtual switch like Open vSwitch (OVS) [107] running on SGW. The traffic filtering rules realise different *isolation levels* for devices based on the security assessment of the device's type. We envisage at least following isolation levels for IoTSentinel, shown in Fig. 5.3:

STRICT This isolation level limits the communication of devices to the *untrusted* group only. No Internet access is allowed for devices in this isolation level.

RESTRICTED In this isolation level, communication of devices is limited to the untrusted group as well as to a limited set of destination addresses on the Internet (e. g., the cloud sevice of the device vendor).

TRUSTED In this isolation level, communication with any devices in the *trusted* group as well as full internet access is permitted.

The security policies for device types in the *restricted* isolation level include also a list of IP addresses or Domain Name System (DNS) names identifying the remote entities with which devices of this type are allowed to communicate with. Typically this set of endpoints would include the addresses to cloud-based back-end services of the device type's manufacturer.

In addition to the aforementioned isolation levels, and division of the network to an untrusted and trusted subnetwork, there is also a *quarantine* network in which newly installed devices are placed before their device type has been identified and an isolation level has been determined. In the quarantine network devices have Internet access but can't communicate with any other devices in either the trusted nor untrusted networks, with the exception of dedicated master devices like the smartphone or laptop of the user which is used to perform the device installation into the user's network.

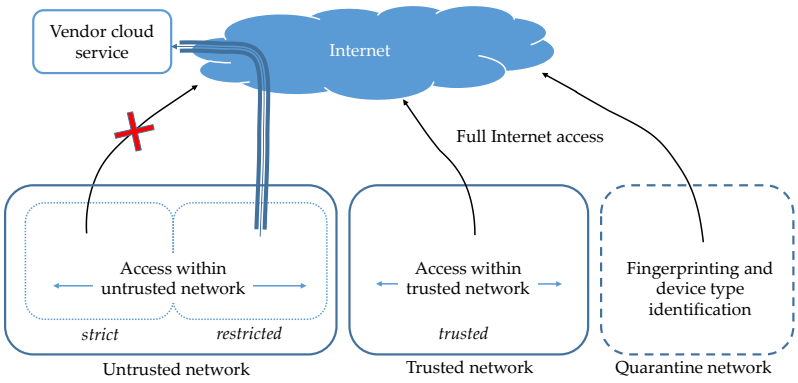


Figure 5.3: Security policy enforcement with IoTSentinel isolation levels

The target of using enforcement rules and isolation levels is primarily to mitigate the risk of device compromise by external adversaries by limiting the communications of potentially vulnerable IoT devices. On the other hand this also mitigates the risk of leaking sensitive user data to an external adversary even if a device in the untrusted group should be compromised, as it is not allowed to communicate with hosts located outside the untrusted virtual network. However, in order to preserve the utility of the device for the user, it may be necessary to allow strictly controlled communications with known benign entities like the IoT device vendor's back-end systems. This is realised through the *restricted* isolation level that enables such dedicated communications.

By isolating potentially vulnerable devices from other devices in the trusted group, we can also protect devices in the trusted group in case a vulnerable device is compromised, e. g., by a local adversary. This way, compromised devices in the untrusted group cannot be used to attack devices in the trusted group and use them to leak sensitive user data or attack targets on the Internet.

In cases in which a device fingerprint **F** cannot be associated with any known device type, the associated device is assigned the *strict* isolation level. Fingerprints associated with such unknown device types can be stored by the IoTSS and re-evaluated when new device-type classifiers are added to the system. If the stored fingerprint matches a new device type, the isolation level of the associated device is updated accordingly and the SGW notified, which will then update the isolation profile of the device. Also, if a new vulnerability associated with a device type is discovered that mandates a change in the device type's assigned isolation level, the IoTSS will notify all SGWs hosting devices of the affected device type. The SGW will then update the isolation profiles of all devices of this type according to the new isolation level.

5.4 EVALUATION

To evaluate the device identification performance of the IoTSentinel framework we developed a experiment set-up to test our approach on real-world IoT devices. We purchased a set of 27 IoT devices and repeated the set-up process in our laboratory network mimicking a typical SOHO network setting typically present in a smart home environment. During the set-up process of each device we collected the data packets emitted by the devices and derived device fingerprints as discussed in Sect. 5.3.2 from them. Using the fingerprints as training data, we then trained the machine-learning models as discussed in Sect. 5.3.3 and tested their performance in identifying the device types of individual devices using classification accuracy as the measure of fitness.

5.4.1 *Experiment Set-Up*

Our evaluation was performed in a lab environment in which an SGW was used tp collect the communication data from tested IoT devices. In our evaluation set-up, we implemented the functionality of the SGW for IoT device measurement collection on a laptop running *Kali Linux* [66]. We used the `hostapd` software package [79] to set up a WiFi AP simulating the WiFi interface of the SGW. An additional external Ethernet interface was connected to the laptop to simulate the Ethernet ports typically present in SOHO WiFi routers.

5.4.1.1 *Data Collection*

The communication packets of the tested IoT devices were captured by running `tcpdump` [137] on the monitored WiFi and Ethernet interfaces. This allowed us to collect all communication packets on both wireless and wired interfaces of the SGW. The data collection was facilitated with the help of a scripted experimentation user interface (UI), providing step-by-step instructions to the test person setting up the device into the laboratory network. These instructions were compiled manually based on the hardcopy manual or on-line instructions provided for the tested devices.

The typical set-up process involves the use of a device- or vendor-specific app, which was installed on a smartphone used to perform the tests. In a few cases the set-up process required the use of a PC application that was installed on a laptop computer used in the tests.

The set-up process, shown in Fig. 5.4, usually involves performing a factory reset (also called hard reset) of the tested device to erase any settings and bringing it to a state identical to when it had not been configured yet. In most cases the device will then set up a WiFi access point to which the user needs to connect the smartphone (1) with the help of the companion app. After this the companion app would

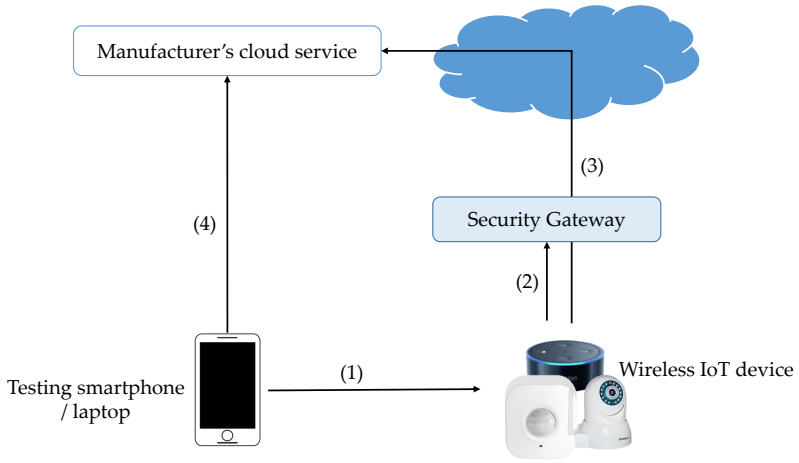


Figure 5.4: Typical set-up process of WiFi-based wireless IoT devices

request the user to provide access credentials to the user's WiFi network (or use credentials previously stored in the companion app) and transmit these to the device being set up. The IoT device will then disconnect from the smartphone and use the provided credentials to connect to the user's WiFi access point (i. e., the Security Gateway in our test set-up) (2). Upon successful connection, the IoT device will typically connect to its manufacturer's cloud service (3), e. g., to check for firmware updates, after which the companion app on the smartphone also is able to monitor and control the device with the help of the cloud-based service (4). During the entire set-up procedure the data packets transmitted by the IoT device were captured and recorded by the SGW for off-line fingerprint extraction and evaluation.

5.4.1.2 *Dataset*

We selected a representative set of IoT devices that were available in the European market in the first quarter of the year 2016 for our experiments. They covered a broad range of functionalities ranging from smart lighting, home automation, and health monitoring to household appliances and IP cameras. Most of the selected devices used WiFi or Ethernet to connect to the user's smart home network. Some devices, particularly such devices that needed to operate on battery power without connection to a power outlet used low-energy protocols like, e. g., *ZigBee* [150], *Z-Wave* [147] or *Bluetooth Low Energy (BLE)* [133]. These devices connected indirectly to the network using a gateway or hub device that was connected via WiFi or Ethernet to the test network. For these devices we focused on monitoring the indirect traffic originating from the hub device when forwarding data from the IoT device to the user's network. A detailed overview of the tested devices and the communication protocols they used is given in Tab. 5.1.

Table 5.1: IoT devices and their connectivity technologies

	DEVICE MODEL	WIFI	ZIGBEE	ETH	Z-WAVE	OTHER
SMART LIGHTING	Osram Lightify Gateway	•	•	○	○	○
	Philips Hue Bridge 3241312018	○	•	•	○	○
	Philips Hue Switch PTM 215Z	○	•	○	○	○
	WeMo Link Bridge F7C031vf	•	•	○	○	○
SMART HOME AUTOMATION	Homematic switch HMIP-PS	○	○	○	○	•
	MAX! Cube LAN Gateway	○	○	•	○	•
	D-Link Hub DCH-G020	•	○	•	•	○
	D-Link Door & Window sensor	○	○	○	•	○
	D-Link Siren DCH-S220	•	○	○	○	○
	D-Link Smart plug DSP-W215	•	○	○	○	○
	D-Link Water sensor DCH-S160	•	○	○	○	○
	D-Link Motion sensor DCH-S150	•	○	○	○	○
	Edimax SP-1101W Smart Plug	•	○	○	○	○
	Edimax SP-2101W Smart Plug	•	○	○	○	○
	Ednet.living power Gateway	•	○	○	○	•
	TP-Link Smart plug HS100	•	○	○	○	○
	TP-Link Smart plug HS110	•	○	○	○	○
	WeMo Insight Switch F7C029de	•	○	○	○	○
	WeMo Switch F7C027de	•	○	○	○	○
HEALTH MONITORING	Withings Wireless Scale WS-30	•	○	○	○	○
	Fitbit Aria WiFi-enabled scale	•	○	○	○	○
HOUSEHOLD APPLIANCES	Smarter iKettle 2.0 SMK20-EU	•	○	○	○	○
	SmarterCoffee SMC10-EU	•	○	○	○	○
IP CAMERAS	D-Link Camera DCS-930L	•	○	•	○	○
	D-Link Camera DCH-935L	•	○	○	○	○
	Ednet IP camera Cube	•	○	•	○	○
	Edimax IC-3115W	•	○	•	○	○

The device set-up experiment was repeated $n = 20$ times for each device to acquire sufficient data to train the device-type-specific classification models used for device identification. From the communication trace of each experiment one device fingerprint was extracted, resulting in a database of 540 fingerprints from 27 different device types.

5.4.1.3 *Results*

The performance of the device identification approach described in Sect. 5.3.3 was evaluated using stratified 10-fold cross-validation, in which the database was divided into 10 equal parts, so-called folds. In each iteration, 9 folds were used as training data to train the device-type classifiers and one fold was used for testing. The cross-validation was repeated 10 times.

The results show very good performance for most IoT device types. For 17 device types the classification accuracy is above 0.95, most of them reaching 1, indicating perfect classification accuracy. However, the average overall classification accuracy is only 0.82, as for 10 devices the classification performance is lower, around 0.5. However, this is still significantly better than a random class assignment which would yield an accuracy of merely $1/27 \approx 0.037$.

Closer analysis of the classification results of the worse-performing 10 device types reveals that the misclassifications of these devices are exclusively occurring between *similar devices of the same device vendor*. For instance, the smart water cooker (*iKettle*) and coffee maker (*SmarterCoffee*) of the device manufacturer *Smarter*[2] receive lower accuracy scores only due to the fact that the device identification approach tends to be confused between these two products. This is caused by the fact that even though these products look very different from the outside, they use an identical hardware and software set-up for realising WiFi connectivity and intelligent functionality. Their communication behaviour is therefore very similar, making it difficult for the identification approach to distinguish between these two device models.

The same applies to the smart power plug devices from *Edimax* and *TP-Link*. Our set of tested devices contains two different smart WiFi power plug models each from both manufacturers: the *SP-1101W* and *SP-2101W* from Edimax and the *HS100* and *HS110* from TP-Link. Also for these device types, misclassification occurs due to a confusion between the particular device models from the same manufacturer. This is easily explained, as the respective power plug models from the same manufacturer are nearly identical devices with very similar form factors: their only difference is that one of the models (SP-2101W from Edimax and HS110 from TP-Link) offers the additional feature of providing real-time measurements about the current

2 https://smarter.am/

flowing through the power plug. It is therefore likely that also for these devices the reason for confusion between the closely related device models is caused by nearly identical hardware and software implementations causing the related device models to display a very similar communication behaviour which is difficult to distinguish by the device identification method.

Also four devices from *D-Link* get confused by the device identification approach: the *Siren DCH-S220*, *Smart plug DSP-W215*, *Water sensor DCH-S160* and *Motion sensor DCH-S150*. A closer look at the firmware versions provide a likely reason for the confusion between devices: by inspecting configuration information of the devices in question we could verify that at least the sensor devices (DCH-S150, DCH-S160 and DCH-S220) indicated to have the same firmware version number. This suggests that the device manufacturer in question likely builds its products based on a generic software platform with only minor device-type-specific modifications to accommodate specific features of the device. This is also supported by recent reports about IoT-specific security vulnerabilities [127] affecting a wide range of different IoT products because they are built using a common software platform with a number of re-used components.

For device identification this means that even though there may be obvious and apparent differences in the features and functionality of some IoT devices, some groups of devices share a large part of their code base in common with other device models in the group. Because of this, their communication behaviour is very similar, making reliable differentiation of devices' types within such groups challenging. This applies especially to communication behaviour during the installation of the device, which IoTSentinel uses for device identification, as the function-specific behaviour of the device is initiated only *after* the device has been successfully installed in the user's home network. As we will discuss in more detail in Sect. 6.2, this could be, however, remedied by extending the device identification process also to the communication behaviour that the devices display after they have been successfully installed in their target environments.

5.4.2 *Prototype Implementation*

We implemented a prototype of IoTSentinel consisting of a *Raspberry Pi 2* [112] development board realising the Security Gateway functionality and an IoT Security Service running on a remote server. The system design of the prototype is shown in Fig. 5.5.

The SGW was realised using a modified version of the open-source-based Floodlight SDN controller [12] supporting the Open vSwitch virtual switch. The Raspberry Pi was configured to provide a wireless AP using the `hostapd` [79] software package.

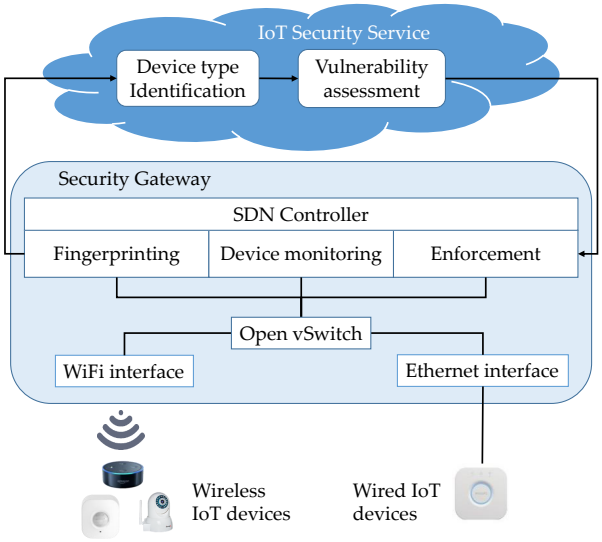


Figure 5.5: Overview of the IoTSentinel prototype design

5.4.2.1 *Device Monitoring*

The device monitoring component is realised as an extension to the SDN controller functionality. This component is responsible for monitoring the MAC addresses of devices connecting to the SGW and initiating fingerprinting if new devices that have previously not been observed join the user's network. This is done by invoking the fingerprinting module and providing to it the MAC address of the device to be identified.

5.4.2.2 *Fingerprinting*

The fingerprinting component uses `tcpdump` [137] to capture packets from the `wlan0` and `eth1` interfaces of the SGW. When receiving a fingerprinting request for a particular MAC address from the device monitoring component, the fingerprinting module will filter packets originating from the given MAC address for a duration of k seconds and initiate feature extraction on the sequence of received packets during this time. As discussed in 5.3.2, the first n packets $p_1, p_2, \ldots, p_n$ will then be used to derive a device fingerprint $\mathbf{F}$ which is then sent to the IoTSS for identification. To reduce the required communication overhead, the fingerprinting component would also first compress the fingerprint before transmitting it to the IoTSS.

In our prototype implementation we used $k = 120\,\text{s}$, to allow for sufficient time for all considered device types' set-up processes to complete within the monitoring period. Based on the evaluation of the collected data, $n = 12$ was determined to be a sufficient number of packets included in fingerprint extraction to provide good classification accuracy.

5.4.2.3 *Device-Type Identification*

The device identification module is implemented on a Linux-based application server using the `scikit-learn` Python library [125] to implement the Random Forest classifiers used in the device-type identification. The identification process as discussed in Sect. 5.3.3 proceeds as follows. After receiving the fingerprint **F** for the to-be identified device from the fingerprinting component, the fingerprint is tested against all Random Forest classifiers (altogether 27 classifiers in our prototype set-up). Each classifier provides a binary prediction indicating whether fingerprint **F** is considered to belong to the device type associated with the classifier in question.

In case more than one classifier provides a positive classification result, edit distance tie-break as discussed in Sect. 5.3.3 is used to determine the final prediction of the device type for fingerprint **F**. The resulting device-type identity is then forwarded to the vulnerability assessment component.

5.4.2.4 *Vulnerability Assessment*

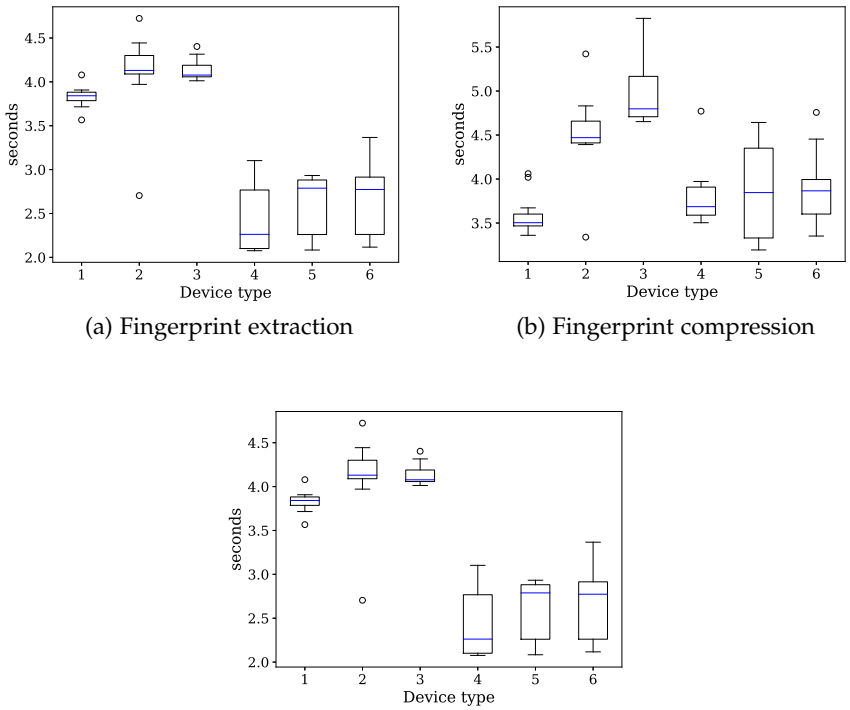
In the prototype implementation, vulnerability assessment is realised with the help of a look-up table associating each device-type identifier with an associated isolation level of *strict*, *restricted* or *trusted* as discussed in Sect. 5.3.5 along with a possible whitelist of IP addresses with which devices are allowed to communicate if they are assigned to the *restricted* isolation level. After retrieving these security policy settings associated with the device type in question the IoTSS will combine these into an isolation profile in the form of a JSON file [61] and send it to the enforcement component of the SGW.

5.4.2.5 *Enforcement*

The enforcement component receives the device type associated with the MAC for which the fingerprinting and device identification was initiated along with the isolation profile from the IoTSS. It parses the profile file and generates necessary traffic flow rules for the SDN controller to realise the intended isolation level for the device. The flow rules are then enforced by the virtual switch controlled by the SDN controller.

5.4.2.6 *Performance Evaluation*

We evaluated the performance of the prototype implementation of IoTSentinel by repeating device-set up experiments ten times each for six different devices representing distinct device types, totalling $n = 60$ different device set-up experiments. In each set-up experiment, data packets from the tested IoT device were collected during



(a) Fingerprint extraction

(b) Fingerprint compression

(c) Device-type classification

Figure 5.6: Time required for fingerprint extraction, compression and device-type identification for different device types (1: *Fitbit Aria*, 2: *Edimax Plug 1101W*, 3: *Edimax Plug 2101W*, 4: *Ednet Gateway*, 5: *TP-Link Plug HS100*, 6: *TP-Link Plug HS110*)

$k = 120\,\text{s}$. We then measured the time required by the prototype system to perform the individual steps of the device-identification process, namely feature extraction, feature compression and device-type classification. The performance of the prototype is shown in Fig. 5.6.

As shown in Fig. 5.6a, the duration for SGW to extract device fingerprints for all device types is 2770 ms to 11 690 ms. The average fingerprinting duration is (4894 ± 1559) ms. Fingerprint compression, shown in Fig. 5.6b, takes on average (4101 ± 627) ms.

For evaluating the device-type identification task we measured the round trip time it takes from the SGW sending the fingerprint $\mathbf{F}$ to the IoTSS and receiving a classification reply. In this evaluation the IoTSS was located in a different country, in more than 1500 km distance. As can be seen from Fig. 5.6c, this nevertheless does not introduce significant delays, as classification takes 2075 ms to 4723 ms, the average time required for the classification step, including communication overheads being (3301 ± 806) ms.

In conclusion, we can say that the performance of the prototype setup is sufficient to support real-world deployment of the IoTSentinel framework. Feature extraction and device-type classification are suc-

cessfully completed in a matter of a few seconds. In comparison to this, the set-up time of an IoT device is typically much longer, ranging up to a few minutes. As the prototype evaluation shows, IoTSentinel allows device-type identification to happen very quickly after a device joins the network of the user, leaving only a very small window of opportunity ($\approx$ 10 s) for an adversary to compromise a vulnerable device before IoTSentinel will apply an appropriate isolation profile to protect it. It can therefore be said that in comparison to the *status quo* IoTSentinel provides significant benefits in making users' networks more resilient against adversarial threats arising from the presence of vulnerable IoT devices in the network.

5.5 SECURITY ANALYSIS

5.5.1 *Masquerading Adversary*

One of our assumptions outlined in Sect. 5.2.2 was that when IoT devices are initially installed in the user's network they are not compromised, i. e., benign. This is a reasonable assumption, as devices typically are in their factory-default state when people purchase them and install them for the first time. Even for repurposed or resold devices the standard practice is to perform a 'hard reset' or 'factory reset' to erase the device's previous state information and settings like access credentials to the device's previous owner's WiFi network and bring the device into a state in which it can be configured to join the new owner's network.

All known major IoT malware so far like *Mirai* [51], *Hajime* [34] or *Persirai* [146] operate in the volatile memory of their host device and do not perform persistent modifications of the installed firmware of IoT devices they infect. As soon as an infected device is restarted or a factory reset is performed, the malicious functionality is removed. The only IoT malware known so far to leave permanent traces of itself is *BrickerBot* [111], which performs a permanent denial of service (PDoS) on its host device by removing or corrupting system files and discontinuing network connectivity. This effectively renders the affected device permanently dysfunctional.

It is therefore highly unlikely that an IoT device that is newly installed into a user network would carry malicious functionality at the time of installation. It will therefore always take some time for the adversary to discover and infect the affected vulnerable device. This should leave sufficient time for IoTSentinel to identify the device in question to take protective measures to mitigate the risk of device compromise as discussed in Sect. 5.3.5.

Should the adversary, however, manage to get the user to install an IoT device it has already compromised, the adversary can try to change the way the IoT device behaves during the device-identifi-

cation process so that it is not identified as belonging to a vulnerable device type and assigned to the *strict* or *restricted* isolation levels, which would make it impossible or very difficult to communicate with the device from the outside and exfiltrate data from it. There are a number of ways the adversary can try to modify the device's behaviour.

5.5.1.1 *Blocking Communications*

The adversary can try to avoid identification of the compromised device by blocking communications with the SGW altogether. However, on the wireless interface, this will also pre-empt the device from joining the network altogether and therefore render the device useless for the adversary. If the adversary allows the device to engage in just enough communications to join the user's network (essentially limiting communications to the WPA2 handshake messages), this will result in a mostly blank device fingerprint $\mathbf{F}$ that cannot be associated with any known device type (Recall from Sect. 5.3.2 that $\mathbf{F}$ will be padded with zeros, if less than n packets are observed during the observation period of k seconds). On the wired Ethernet interface, the device identification process will be triggered as soon as the device sends its first data packet. If the device does after that not send any packets during the following k seconds during which the SGW extracts the device fingerprint, the outcome will also be a mostly blank fingerprint that cannot be mapped to any known device type.

In both cases, as discussed in Sect. 5.3.5, the device is treated to be of an unknown device type and will consequently be assigned the *strict* isolation level, making it impossible for the adversary to communicate with the compromised device or exfiltrate sensitive data from it.

5.5.1.2 *Spoofing MAC Address*

If the adversary manages to compromise a vulnerable device, it can try to spoof the device's MAC address in order to masquerade as another device in the network. This has, however, the prerequisite that the network interface card (NIC) of the device must support the modification of the MAC address. For wired devices connected to the SGW over Ethernet MAC spoofing can be easily mitigated, as the device is connected to a dedicated port on the SGW and a change in the MAC address of the connected device can be detected in a straightforward way. For devices connected over WiFi, the adversary can choose two options: spoofing an unknown MAC address, or, spoofing an existing MAC address of a device belonging to the *trusted* isolation level.

If the adversary spoofs a MAC address that is previously unknown to the SGW, this will trigger the device identification process to be initiated for the device. The adversary must then be able to masquerade

as a legitimate device of a device type associated with the *trusted* isolation level as discussed in Sect. 5.5.1.3 by mimicking the set-up process of this device. This is necessary to get the spoofed MAC address assigned to the *trusted* isolation level. Until the identification process is completed the device will remain in the *quarantine* network and will thus be unable to communicate with other devices in the user's network.

The adversary can also spoof a MAC address that is already known to the SGW and belongs to the *trusted* isolation level. Such spoofing can, however, be detected by the SGW if there is an association with the genuine device owning this MAC address that is still valid and in use. As each association using WPA2 is characterised by a unique Pairwise Transient Key (PTK) [60], it is possible for the SGW to notice the presence of two different devices attempting to use the same MAC address at the same time.

5.5.1.3 *Masquerading as Another Device Type*

The adversary may attempt to masquerade a compromised IoT device as a different device belonging to a device type belonging to the *trusted* isolation level in order to circumvent the traffic restrictions imposed on the device by the SGW. In order to succeed in this, the adversary must mimic the communication behaviour of this device during its set-up phase.

To do this the adversary must know the exact behaviour of the intended device beforehand and program the vulnerable device to communicate according to this behaviour profile. However, this brings a practical difficulty to the adversary, as the typical set-up process of an IoT device in many cases requires the direct involvement of the user and the use of a device-type-specific set-up application on the smartphone of the user. In order to be able to mimic the communication behaviour of a different device type, the adversary would need to fool the user into using the set-up app of the spoofed device type instead of the real device's set-up app in order to maintain the communication semantics between the IoT device and the smartphone used to set it up.

It is in principle possible that the adversary could try to fool the system by merely replaying a set of data packets mimicking a set-up session with an imaginary, non-existent set-up app, i. e., by replaying only messages originating from the IoT device. However, this can be easily detected by the SGW. Since the adversary can spoof only messages originating from the compromised IoT device, but is not able to spoof the corresponding messages coming from the set-up app on the user's smartphone, it is relatively easy for the SGW to notice that these messages are missing and raise an alarm by notifying the user. In the meantime the concerned device would be placed in the *strict* isolation level.

5.5.1.4 *Unknown Vulnerabilities*

It is possible that an IoT device has a vulnerability that is not yet known to the IoTSS when the device is added to the user's network and its device type is identified by IoTSentinel, resulting in the device being—incorrectly—placed into the *trusted* isolation level. During this time an adversary may be able to exploit the vulnerability and compromise the device. However, as soon as the vulnerability of this device type is discovered and published, e. g., in a public vulnerability repository, IoTSS will update the isolation level for the affected device type and push this update to all SGWs that host devices of this type, resulting in devices of this type to be moved into the *strict* or *restricted* isolation level.

This can even happen *before* the actual vulnerability is discovered, e. g., when security incident reports start showing a significant correlation with the device type in question. As a pro-active measure, the affected device type can therefore be moved to the *strict* or *restricted* isolation level until IoTSS manages to resolve whether the security incidents are caused by a vulnerability in the affected device type or not. Should the determined reason prove to be not related to vulnerabilities in the device type, IoTSS can simply revoke the pro-active measure by assigning the device type back to the *trusted* isolation level.

The window of opportunity for the adversary to exploit previously unknown vulnerabilities is therefore relatively small, and requires the adversary to be aware of a vulnerability that is not yet generally known. Also, massive exploitation of such vulnerabilities can be quickly counteracted by the IoTSS by distributing the updated isolation profile for the affected device type as a pro-active measure, as discussed above.

5.5.2 *Impact of Device Mis-Classification*

As discussed in Sect. 5.4.1.3, for most IoT device types the classification results are very accurate. However, within particular groups of device types from the same manufacturer, devices are confused within said groups, leading to lower nominal classification accuracy. This seems to be caused by the fact that devices within the group are based on a common code base with shared software components, causing their communication behaviour especially during the installation to be very similar and making distinguishing between individual device types within the group difficult.

However, as recent incidents show [127], security vulnerabilities often impact all devices that share the code containing the vulnerability. From a security point of view, it is therefore not necessary to identify the exact make and model of each device, but it is sufficient to identify whether the device belongs to a group of devices that is known to share a vulnerability. As we discuss in more detail in Sect. 6.2, it

might therefore in the future be more useful to focus device identification on such clusters of device types showing identical behaviour instead of trying to pinpoint the exact device-type label given to a device by the manufacturer.

5.5.3 *Attacks Against IoTSentinel Components*

As discussed in Sect. 5.2.2, our adversary model makes the specific assumption that the components of the IoTSentinel system are trustworthy, i. e., not compromised by the adversary. However, it is thinkable that an adversary would attempt to utilise parts of the IoTSentinel system it has access to for attacking the system itself. In order to do this the adversary has a number of options.

5.5.3.1 *Physical Attacks Against Security Gateways*

The Security Gateway (SGW) is typically located within the premises of user households. Therefore it is thinkable that a malicious user could physically compromise it in order to take it under its control. Unless the SGW is equipped with specific tamper-resistant hardware capable of verifying the integrity of the SGW using, for instance, remote attestation (e.g., [28]), we need to assume that an adversary with physical access to the SGW will be able to modify the functionality of it at will. Since this type of attack requires physical access to each SGW it wants to compromise, it is very difficult to scale to large numbers of SGWs. If we denote the total number of SGWs in the system with n and the number of SGWs compromised by the adversary with k, we can assume that in the case of physical attacks $k \ll n$.

5.5.3.2 *Software Attacks*

In a software attack, a remote adversary uses possible vulnerabilities in the implementation of the SGW itself to infiltrate and take over control over it. This type of attack is potentially more scalable, as the adversary does not need physical access to the SGW in order to be able to compromise it. Thereby the number k of compromised SGW in relation to the total number n of SGWs is higher. However, we assume that an adversary will not be able to compromise a *majority* of the overall security gateway population, i. e., $k < \frac{n}{2}$.

5.5.3.3 *Attack Impact*

In both aforementioned attacks, the adversary can gain the possibility to disable or manipulate the operation of the SGW. By doing this, it can render any IoT devices protected by the SGW vulnerable to attacks. However, the impact of the attack is limited to each local network associated with the SGW.

In case of physical attacks this means that it is a malicious user himself attacking his network, whereby rendering his devices unprotected is something the user is deliberately causing, a case equivalent to a user on purpose disabling his firewall or virus scanner, whose purpose is to protect the user's devices. From the system point of view, such attacks are therefore causing harm mostly to the attacker itself. The only motivation for an attacker to engage in such attacks is therefore limited to cases in which the adversary tries to use a compromised SGW against the IoTSentinel system itself, as outlined in Sect. 5.5.4 below.

To remedy attacks, the system can employ remote attestation techniques, as mentioned above, to ensure that the operational status of the software executing on the SGWs is as specified. This requires, however, trusted hardware support on the SGW, which may not be available in practice. In such cases the system can be equipped with extensive self-monitoring components that would regularly inspect the operating state of individual SGWs and sample their reported operational characteristics. Potentially malfunctioning SGWs could then be identified by detecting deviating behaviour of the affected SGWs. While this approach does in the absence of remote attestation techniques not provide absolute assurance of detecting malicious nodes, it does, however, raise the bar for successful attacks, as the adversary would need to be able to mimic the normal operational characteristics of the SGW in order to remain undetected, thereby raising the cost of the attack.

5.5.4 *Data Poisoning Attacks*

An adversary may attack against the IoTSentinel system by trying to manipulate the device identification process. It can attempt to do this with the help of *data poisoning*, i. e., by influencing the inputs based on which the IoT Security Service (IoTSS) builds the classification models used in device-type identification. The adversary has two ways how to achieve this: providing manipulated communication patterns to the SGW, or, compromising the SGW and providing manipulated device fingerprints to the IoTSS.

5.5.4.1 *Manipulating IoT Device Communication Patterns*

To manipulate the communication patterns associated with an IoT device, the adversary needs to compromise the device and modify its communication behaviour. However, as discussed in Sect. 5.2.2, we assume that new device models, when entering the market and being deployed, are initially benign. It will take considerable time by adversaries to compromise the device and invoke their malicious functionality. The initial device fingerprinting performed by IoTSentinel is relatively short, i. e., 2 min. This leaves in practice too little time for

the adversary to compromise the device before its device fingerprint is extracted and used to build the device identification model for the device-type.

Another hindrance for the adversary is that in order to influence the training of the device-type identification model, it needs to be able to manipulate a sufficiently large number of device fingerprints in order to influence the resulting identification model. This means that after the initial introduction of a new IoT device type the adversary needs to be able to very quickly compromise a significant fraction of these devices *before* the SGWs hosting these devices have a chance to extract benign device fingerprints for these devices. Since this happens within a very limited time frame (2 min), it is very difficult for an external adversary to locate and compromise a significant fraction of the devices being introduced to various local networks all over the Internet during such a short period of time.

5.5.4.2 *Manipulating Device Fingerprints*

If the adversary is able to compromise the SGW as discussed above in Sects. 5.5.3.1 and 5.5.3.2, it can produce entirely *fabricated* fingerprints to the IoTSS in order to corrupt the training of the device identification model. As in the case of manipulating fingerprints, the adversary needs to be able to inject a sufficiently large number of fabricated fingerprints in order to corrupt the learnt model. In this case the adversary has a larger time window during which it can act, as it is not limited by the 2-minute time window during which SGWs aggregate the device fingerprint, but can fabricate the fingerprint at a time of its choosing. Nevertheless it must do so before a sufficient number of benign device fingerprints of a newly introduced device model are provided by benign SGWs, and the device identification model is trained by IoTSS. The effectiveness of this attack is therefore highly dependent on how many SGWs the adversary is able to bring under its control.

In the case of physical attacks against the SGW, as discussed above in Sect. 5.5.3.1, the number k of SGWs that the adversary can compromise in relation to the overall number n of SGWs in the system does not easily scale to substantial numbers, i. e., $k \ll n$, limiting the impact that the adversary can have. It will therefore be challenging for any adversary to stage effective fingerprint manipulation attacks that utilise attacks based on physical compromise of SGW.

Software attacks, as discussed in Sect. 5.5.3.2, however, can potentially be more scalable, as the adversary doesn't require physical access to the targeted SGW, but can stage the attack remotely, thereby being able to simultaneously target a larger population of SGWs. By compromising a large subset of SGWs, the adversary could fabricate a sufficiently large number of device fingerprints to effectively corrupt the trained device-type identification model. However, the more SGWs

the adversary compromises, the higher is also the likelihood that malicious actions of the adversary are detected and countermeasures initiated, e. g., by identifying the vulnerabilities leading to SGW compromise and issuing security patches to the SGWs, thereby eliminating adversary control and making them resilient against attacks targeting the vulnerabilities in question.

5.5.4.3 *Organisational Measures Against Poisoning Attacks*

The above discussion about data poisoning attacks is valid under the assumption that all SGWs contribute device fingerprints to the training process of new device-type classification models in the same way. However, additional organisational measures can be taken to improve the resilience of the system against data poisoning attacks.

One possible approach could be to apply pre-screening of data contributing to the training dataset. This could be achieved by admitting only such fingerprints to the training dataset that have been generated from communication traces collected under controlled settings, e. g., in a dedicated testing laboratory of the IoTSS itself.

While such controlled fingerprint collection would most likely not be feasible for all IoT devices in general due to the enormous number of different IoT device models on the market, it could nevertheless be performed for the most popular devices. This would allow to cover a major fraction of devices with trusted and reliable detection models. The utility of less prevalent IoT device types for which trusted detection models may be unavailable is from the point of view of an adversary much more limited, as such devices are fewer in number and therefore more difficult localise and exploit by attacks.

5.5.4.4 *Poisoning Mitigation Measures*

Defences against adversarial machine learning attacks like data poisoning is currently a lively research topic. Approaches to mitigating such attacks rely, e. g., on eliminating the effect of outlier (i. e., malicious) data points from the training dataset [116], or, identifying data clients providing manipulated training data [129] in a distributed learning setting.

In IoTSentinel, such mitigation measures need to be applied at the IoTSS, as SGWs do initially not have any information about the distribution of raw input data concerning a specific device type, since the device type is at training time new to the whole system. Applying outlier elimination at the SGW is therefore not possible. Moreover, detecting device fingerprints provided by potentially compromised SGWs will in any case be possible only at the IoTSS. In the IoTSentinel setting, therefore, the focus of poisoning mitigation measures needs to be in identifying manipulated device fingerprints (generated either by IoT device communication manipulation (Sect. 5.5.4.1), or, dir-

ect manipulation (Sect. 5.5.4.2)) before they are used in training the device classification model.

Detecting manipulated device fingerprints can be performed by adopting the approach presented by Shen *et al.* [129]: device fingerprints provided by different SGWs are first collected and clustered into two clusters. As long as the adversary won't be able to compromise a majority of the SGWs, or compromise more than half of the devices of a particular device type (which is a reasonable assumption), we can assume that the cluster with the majority of device fingerprints represents benign fingerprints, i. e., fingerprints that have not been manipulated by the adversary. The learning of the classification of the device type can then be limited to device fingerprints belonging to this majority cluster in order to eliminate manipulated device fingerprints from the learning process.

Adversarial machine learning is a research area that is currently the target of lively research activities looking at ways to attack machine learning models used, e. g., for speech and image recognition. In line of these activities, our future research will explore further the problem of data poisoning in systems like IoTSentinel and develop and evaluate concrete defence mechanisms for mitigating such attacks.

5.6 RELATED WORK

5.6.1 *Device Fingerprinting*

A number of device fingerprinting approaches for identifying devices based on their communication characteristics have been proposed.

5.6.1.1 *Protocol-Based Approaches*

Bratus *et al.* [15] propose an approach in which the the combination of WiFi chipset and driver of a 802.11 wireless device can be determined through active probing of the device's wireless interface using specially crafted 802.11 protocol frames. As the responses of different devices to particular probe frames will differ, their solution uses a decision tree to identify the WiFi chipset and driver of the device (Bratus *et al.* consider primarily WiFi APs) based on the observed responses to a specific sequence of probe frames sent to the device. Cache presented a passive fingerprinting approach [20] that is based on examining the duration field in 802.11 frames to identify the WiFi driver implementation in question. The rationale for using this feature as the basis for device fingerprints is based on the observation that the duration field assumes only a few distinct values for different packet types depending on the driver implementation, allowing it to be used for identifying the particular driver. For the purpose of *device-type* identification the approaches above are, however, not suitable, as the same chipset and driver may be shared among many different

kinds of device types, making distinction between individual devices or device types impossible.

To enable identification of *individual devices* sharing identical wireless chipset hardware and drivers, Maurice *et al.* [83] therefore introduced an extension to Cache's fingerprinting scheme. The extension is based on injecting a limited number of raw 802.11 frames with randomly-selected duration fields in order to make distinction between individual devices feasible. Their approach requires, however, installation of a daemon on all devices to be identified, as it relies on active injection of frames by the end-devices and is therefore not applicable to IoT scenarios in general. It can also not be used for device-type identification, as it only allows to distinguish individual devices with the same wireless chipset and driver, giving no information as to the *type* of device in question.

5.6.1.2 *Packet Timing-Based Approaches*

Another approach employed by Franklin *et al.* [40] used observations related to the time intervals between 802.11 probing frames emitted by wireless interfaces of a device to identify the wireless device driver used by the device. The identification is based on the observation that WiFi drivers implement different scanning algorithms, resulting in measurable differences in the timing patterns of probing frames, which in turn can be used for identifying the device driver in question.

5.6.1.3 *Clock Skew-Based Approaches*

Other works like the ones by Kohno, Broido and Claffy [70] propose to use a hardware-specific property like clock skew for identifying individual devices. The advantage here is that clock skew is a property that is specific to the individual hardware instantiation of the device and it is observable from the communication trace of the device in question. Whereas the approach by Kohno *et al.* utilised the Transmission Control Protocol (TCP) timestamp option and Internet Control Message Protocol (ICMP) timestamp requests to obtain measurements for clock skew evaluation, Jana *et al.* [62] apply the approach using Time Synchronization Function (TSF) time stamps of WiFi beacon and response packets in a local setting. In a follow-up work, Arackaparambil *et al.* [4] identified potential vulnerabilities in the way Kohno *et al.* and Jana *et al.* measure clock drift and proposed countermeasures to detect adversarial APs attempting to spoof the clock drift of other, legitimate APs.

5.6.1.4 *RF-Fingerprinting-Based Approaches*

Other approaches for device identification have proposed to use Radio Frequency (RF) fingerprinting. Ureten and Serinken [141] invest-

igated the use of amplitude profiles of RF signals for device identification, being able to distinguish individual devices even having identical wireless transmitter hardware. Another approach taken by Brik *et al.* [17] utilises radiometric measurements based on modulation errors caused by inherent physical imperfections of network interface cards (NICs) to identify individual wireless devices. The main application area for the proposed hardware-based device-identification methods is the possibility to identify possible adversarial rogue APs without the need to support cryptographic protocols for AP authentication.

5.6.1.5 *Software-Based Approaches*

Also software-based approaches for device identification have been proposed. A study by Pang *et al.* [104] showed that in many cases characteristics of WiFi traffic like sets of network destinations, SSID probes, broadcast packet sizes and MAC protocol fields can be used to uniquely identify the users of devices emitting the traffic. Kurtz *et al.* [72] used configuration information about device settings to identify mobile devices and their users.

5.6.1.6 *Sensor-Based Approaches*

Bojinov *et al.* [13] demonstrated how device-specific sensing artefacts in measurements made by sensors on mobile devices could be used used to derive device-specific fingerprints for identifying individual mobile devices. Bertini *et al.* [11] present an approach for using the inherent noise pattern generated by the imaging sensor of smartphone cameras to distinguish the origin device of images taken with the camera. Van Goethem *et al.* [142] demonstrated that characteristic features of accelerometer readings on mobile devices could also be used to uniquely distinguish between individual mobile devices. Sharaf-Dabbagh and Saad [128] introduced a framework for using statistical modelling of context fingerprint values provided by IoT devices for identifying whether the values originate from a legitimate device or a masquerading attacker.

5.6.2 *Device-Type Fingerprinting*

One of the very few approaches addressing the problem of *device-type identification* instead of identifying individual devices has been presented by Gao *et al.* [42]. In their approach they seek to identify the type of an AP by performing black-box testing on it by observing how it processes a large sequence of network packets. The characteristic fingerprint of the AP is based on observing the time shifts in the inter-arrival time (IAT) of individual packets. These are influenced by how the packet sequence is processed by the internal implementa-

tion of the AP and can therefore be used as characteristic features for identifying the type of AP. However, the approach of Gao *et al.* is limited to router-type devices performing packet-forwarding. Their approach is therefore not applicable to the vast majority of IoT devices in general.

Another approach for device-type identification (along with device identification) is *GTID* by Radhakrishnan *et al.* [110]. Their fingerprinting approach is based on feature vectors derived from the IAT of observed packets of particular traffic types, e. g., `ping`, `scp`, or *Skype*. In contrast to IoTSentinel, GTID uses only a single artificial neural network (ANN) based multi-class classification model to identify devices and device types. It requires also a significant amount of observed packets to construct feature vectors for model training and device-type and device identification. In the GTID scenario this is not much of a problem as it addresses primarily general-purpose computing devices like laptops, tablets and smartphones that naturally generate significant amounts of data traffic. In many IoT scenarios, however, IoT devices generate only very little network communications, making it challenging to apply a similar approach on IoT devices. Also, the use of only one single classification model accommodating all device types poses practical challenges towards re-training of the model when new device types emerge, in contrast to the multiple classifiers-approach taken by IoTSentinel.

5.6.3 *Device Authentication in IoT*

A number of schemes for device authentication in IoT have been proposed taking the specific requirements related to the resource-constrained nature of many IoT devices into account. For example, Hernández-Ramos *et al.* [54] present and extension of the EAPOL standard authentication framework tailored that specifically considers the resource constraints related to IoT devices. In addition to the context-based authentication work presented in Chap. 4, Zenger *et al.* [148] present a scheme for proximity-based authentication of devices which is based on measurements of the location-based channel randomness between an access point and two IoT devices. Other approaches like the one presented by Mora-Afonso *et al.* [97] propose to use a location-limited channel like NFC and Identity-Based Cryptography (IBC) for secure authentication of devices.

While all of these approaches provide solutions for secure authentication, they do not directly address the problem of *device-type identification*, which is a core requirement for proactive security solutions proposed by IoTSentinel in the form of network isolation of known vulnerable devices. These approaches are also not suited for our scenario involving a *brownfield* landscape of legacy IoT devices, as all of

them require dedicated components to be installed on the involved devices.

5.6.4 *Run-Time Security Enforcement*

To assure the security of the run-time behaviour of IoT devices a number of approaches have been proposed. The *SIFT* framework of Liang *et al.* [75] seeks to achieve this by providing a safety-centric programming framework for assuring that conflicts or safety policy violations do not occur between applications run on IoT devices. This approach is, however, only applicable to the development of new IoT applications, not already-deployed IoT systems.

One of the first works to address intrusion detection specifically in IoT systems is the *SVELTE* by Raza *et al.* [113]. Their system targets low-energy nodes on IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN) using a hybrid approach utilising both signature- and anomaly detection-based for detecting intrusions. At the time of writing, however, the majority of IoT devices on the market utilise other networking technologies than 6LoWPAN, therefore limiting the practical applicability of their approach.

Another approach presented by Gisdakis *et al.* [44] is targeted at participatory sensing applications. Their system, entitled *SHIELD*, uses data verification techniques to identify and eliminate malicious participants in a participatory sensing system. While such approaches could be used to validate information in a network of IoT devices, it is applicable only to individual applications with a well-defined scope and semantics. Its applicability to IoT devices and applications in general is, however, limited due to the vast heterogeneity of IoT devices and their supported functionalities.

In comparison to these systems the approach taken by IoTSentinel is somewhat different. While the aforementioned approaches seek to identify malicious behaviour or falsified data during the system's operation, IoTSentinel follows a pre-emptive strategy, seeking to mitigate security threats posed by the presence of vulnerable devices in the network by limiting permissible network communications of these devices so that the adversary is not able to exploit the vulnerabilities potentially present in targeted IoT devices and compromise them.

5.6.5 *Commercial IoT Security Solutions*

Recently Internet security firm *F-Secure*[3] introduced a commercial product called *F-Secure SENSE* [36] which is an 'intelligent router' aiming at protecting end-users' home networks against IoT-related threats. SENSE focuses, however, on traditional protection vectors

3 https://www.f-secure.com

like anti-virus capabilities, blocking botnet traffic and malicious websites [144]. It does not seem to have capabilities for identifying arbitrary IoT devices based on their communication behaviour.

5.7 SUMMARY AND CONCLUSION

In course of the rapid expansion of the so-called Internet of Things many new device manufacturers are bringing novel network-enabled products to the market. In many cases, the focus of these players is on fast deployment of their products, leaving only little time and resources to design and implement sound security architectures for their products. Often, manufacturers are also not able to publish adequate security updates for their products in a timely manner, or, appropriate facilities for automatic application of security patches are lacking. This often leads to a situation in which IoT devices with security vulnerabilities are present in users' home networks, leaving them susceptible against attacks for significant amounts of time. The recent emergence of so-called *IoT malware* is exacerbating the threat posed by vulnerable devices to the whole IoT ecosystem.

To encounter the threat of vulnerable IoT devices, we presented IoTSentinel, a security framework for identifying vulnerable IoT devices and enforcing appropriate network isolation policies in order to protect vulnerable devices from being compromised, and, in case devices are compromised, limit the amount of damage they can inflict on other devices in the user's network. IoTSentinel does so by identifying the types of devices newly added to the user's network and enforcing network traffic filtering for devices known to have security vulnerabilities. In contrast to earlier work on device identification, the focus of IoTSentinel is on identifying *device types*, as security vulnerabilities typically affect whole classes of devices and not only individual devices. The device identification mechanism in IoTSentinel is tailored to the requirements of IoT networks, where many devices typically do not emit significant quantities of network traffic, making the use of previous device-type identification approaches difficult, if not impossible.

IoTSentinel therefore employs a device fingerprinting approach that extracts characteristic fingerprints from the first packets an IoT device sends during its set-up process in the network. Since communication behaviour of a device during its set-up is quite characteristic, it can be used to distinguish device types, as our evaluation in Sect. 5.4 shows. Based on the information of a device's type, IoTSentinel will employ device-specific isolation policies limiting network traffic to devices that are known to have vulnerabilities. This mitigates the risk of external adversaries exploiting these vulnerabilities to compromise the device and use it for attacks against other devices in the user's

network or the Internet, or, exfiltrate sensitive user information from the device.

A major benefit of the approach followed in IoTSentinel is that it is applicable to so-called *brownfield* scenarios, i. e., it can be deployed in environments with legacy devices that do not offer the possibility to update device software or instrument the behaviour of devices. IoTSentinel does not have specific requirements towards the IoT devices installed in the network and is therefore applicable to a large majority of IoT devices available on the market today.

6 DISCUSSION AND CONCLUSION

6.1 SUMMARY OF DISSERTATION

In this dissertation we have investigated approaches utilising context and communications profiling for the purpose of security and privacy management in mobile and IoT environments. The main motivation for our work has been the observation that the ever-increasing number and complexity of applications and devices that need to be configured to protect the privacy of users and the integrity and confidentiality of sensitive information is rapidly growing, mandating the development of new approaches for managing them. In particular, conventional approaches in which security and privacy management is based on pre-defined settings or policy configurations is quickly coming to its limits. On one hand, the sheer number of settings that needs to be managed by users is becoming so large that it is not realistic to assume that users would be willing to invest the required effort and time for setting up, refining and continuously managing their policies. On the other hand, the currently ongoing rapid growth of the IoT device market makes it in practice impossible to base security management on pre-defined security settings, as there are new IoT devices being brought to the market by a myriad of different device manufacturers. This makes it very challenging to obtain and maintain a comprehensive set of security settings for all possible IoT devices, as there are constantly new, previously unknown device types coming to the market.

To encounter these challenges we advocate the use of an approach in which profiling and machine learning-based techniques are used to model the relevant environment and entities and automatically *learn* appropriate settings to be applied in particular situations or for particular entities. In this dissertation, we investigate the use of profiling-based approaches in four distinct use cases applying profiling approaches on both *contextual information* and *communication behaviour*.

6.1.1 *Context Profiling for Policy Adaptation*

In chapter 2, we investigated the use of context profiling approaches to adjust the security and privacy policy settings of a mobile device dynamically based on the particular *context* in which the user (and consequently her mobile device) are located in. We showed that by utilising contextual measurements obtained with a mobile device's context sensors about the location, surrounding WiFi access points, as

well as Bluetooth devices, we could profile the *familiarity* of locations and the persons present. We developed an approach in which we utilised profiled information about the user's contexts together with limited user feedback about the perceived security- and privacy-relevant properties of the context to train a machine-learning based model that is capable of making predictions about the security and privacy risks pertinent to particular contexts. In our use cases we then utilise these predictions to adjust the device locking time out as well as third-party applications' access to sensor information of the mobile device in order to protect the user from threats arising from the misuse of her device by unauthorised parties, or, infection by so-called *sensory malware*. We demonstrated that by applying such a context-profiling approach, we can successfully strike a balance between improved user experience due to reduced need for explicit user involvement in making enforcement decisions and sufficient protection against security risks.

6.1.2 *Context-Based Proofs-of-Presence*

We further showed in chapter 3, how a context-profiling approach can be used to provide dynamic proofs of presence in use cases related to Location-Based Service (LBS) and presence sharing. The approach utilises the fact that devices present in the same context can use mutual measurements of their ambient context as a proof for presence in the same ambient environment. By sending its context measurements to a verifier a prover can prove to the verifier that it is actually located in the same context as the prover. Proofs-of-presence are required to encounter the risk of *location cheating attacks*, in which a dishonest adversary wants to make an LBS or another peer to believe it is present in a specific location, while it in reality is not. We showed also that a straightforward use of context measurements is susceptible to *context guessing attacks* particularly in contexts in which the context is mostly static and therefore does not contain sufficient entropy, so that it is relatively easy for an adversary having historical information about the context to *fabricate* context measurements that will with high likelihood be accepted by the verifier as genuine. To mitigate such context-guessing attacks, we therefore introduced an approach utilising the *surprisal* of context measurements to filter out such context-based proofs-of-presence that are too easily guessed by an adversary and thereby providing improved resilience against context-guessing attacks.

6.1.3 *Context-Based Authentication*

We discussed a related but different application for measurements of the ambient context in chapter 4, in which we discussed the use of

context measurements for device authentication. We developed an iterative approach in which pairing counterparts present in the same contextual environment, e. g., in the same room gradually increase their belief in the counterpart's authenticity by repeatedly performing context-based authentications based on *context fingerprints* that both devices observe. The context authentication is based on extracting fingerprint bits from changes in observed quantities of particular context modalities like the ambient noise level. Using an error-correcting code a device can enable a pairing counterpart that is in possession of a fingerprint that is similar to the device's fingerprint to retrieve a shared pairing secret that is needed for successful context authentication. However, any counterpart that is not in possession of a sufficiently similar fingerprint will not be able to retrieve the secret. Contrary to previous approaches proposing similar schemes utilising error-correcting codes, we present a thorough security analysis of such schemes, taking relevant factors like the entropy loss incurred by the used error-correction code and the inherent entropy rate of the context into account.

6.1.4 *IoT Security Management Based on Communications Profiling*

Finally, we introduced a framework for automated device-type identification for IoT for the purpose of security management of local IoT networks in chapter 5. The approach is based on monitoring the communications of IoT devices that are newly introduced in the network and deriving a *device fingerprint* from it, describing the communication behaviour of each device. Utilising a machine-learning based approach the fingerprints are classified into the respective device type that the device belongs to. The rationale for this is that by identifying the device's type, appropriate mitigation measures can be taken for such devices that belong to a device type with known security vulnerabilities. Such mitigations can comprise, e. g., isolating devices that are known to be vulnerable in order to protect them from being compromised, and, to protect other devices in the network from adverse effects in the case that the affected device should be successfully compromised. We evaluate the presented device-type identification approach based on real-world data and show that since typical IoT devices are single-use appliances, their behaviour is limited and follows characteristic behaviour patterns that allows for accurate identification of specific device-types in most cases.

6.2 FUTURE RESEARCH DIRECTIONS

6.2.1 *Communicating Inferred Security and Privacy Settings to Users*

As mentioned in Sect. 2.7, the ConXsense framework introduced in chapter 2 aims at automating policy decisions in order to make security policy management easier and more user-friendly for regular users who are not willing to spend a lot of time in defining and maintaining their policies. While this approach carries with it the promise of making policy management significantly more convenient for the user, it also bears the risk that the perception of the user of what she thinks the system is doing and the actual decision making of the the system drift apart, leading to situations in which the system behaves in (from the user's point of view) unexpected ways, potentially leading to undesired outcomes. It is therefore very important to develop in parallel with the deployment of automated reasoning systems like ConXsense also approaches for communicating internal state and the reasons *why* the automated policy decision making system is taking particular decisions and based on which information. Only by openly communicating about the decision making process does it become understandable to the user and allows her also to take possible corrective actions in an informed way. Developing and rigorously evaluating such measures for user interaction and visualisation of the system's internal state remains an interesting future research challenge.

6.2.2 *Practical Considerations for Context-Based Authentication*

In chapter 4 we outlined the theoretical constraints that need to be taken into account when evaluating the security of context-based authentication. These constraints relate to the inherent entropy rate provided by the context as well as the level of contextual separation present in the deployment environment. Both of these factors affect the selection of the error-correction level of the used ECC as well as the required duration of pairing. In our evaluation we determined these factors through empirical measurements in the experimental setting.

For the practical deployment of the presented approach, however, a comprehensive framework for estimating and measuring these contextual constraints is required. This framework needs to be based on the analysis of large-scale empirical measurements in a wide range of typical deployment settings of IoT devices in different environments. The result would be a set of practical guidelines assisting implementers of context-based authentication approaches to select appropriate parameters so that the resulting authentication solution is secure while providing maximal utility in terms of time needed for pairing and computational and communication overhead.

6.2.3 *Extension of Device-Type Identification to On-Line Device Behaviour*

The device-type identification approach presented in chapter 5 relies on monitoring device behaviour during the *install time* of each new IoT device being introduced to the system. While having been demonstrated to be effective, this approach has the drawback that it is not applicable to legacy installations with devices already present in the target network. An obvious extension to the work of this dissertation is therefore to augment the device-type identification approach to consider not only the install-time behaviour of devices, but also their on-line behaviour during normal device operations after they have been installed to the network. This new aspect to device-type identification is part of our ongoing research work.

6.2.4 *Enhanced Notions of Device Types*

The evaluation of our device-type identification approach in Sect. 5.4 shows that some device types are easily confused by the identification approach. This happens primarily between devices of the same manufacturer with very similar hardware and software configurations. This raises therefore the question, whether a straightforward definition of device type targeted at identifying the exact model name given to the device by its manufacturer is actually useful when security management is the main target for device-type identification. Indeed, in our ongoing research work we intend to adopt a more abstract notion of device type. As security vulnerabilities are closely linked to specific hardware and software configurations, from a security management point of view it is sufficient to identify a device type at the level of such configurations, and not at the level of manufacturer-provided model names. In our ongoing research we have embraced this approach and are thus able to achieve better accuracy with regards to device-type identification.

6.2.5 *Augmenting Proactive with Reactive Defences*

The primary goal of the IoTSentinel system is to provide proactive defence mechanisms for protecting networks with vulnerable IoT devices. A natural extension of this approach is to augment these defences with reactive measures like intrusion detection. We believe intrusion detection approaches can greatly benefit from the capability to automatically identify the type of devices, as it allows to tailor detection models aimed at detecting changes in communication behaviour caused by adversarial compromise to each device's identified type. This has the promise of making the detection process more accurate and avoid false alarms. Initial results from our ongoing work suggest this to be the case.

7

ABOUT THE AUTHOR

Markus Miettinen (born on 20.05.1976 in Helsinki, Finland) received his M.Sc. in computer science in 2002 from the University of Helsinki, Finland. From 1999 to 2012 he worked as a trainee, research engineer and senior researcher at the Nokia Research Center in Helsinki, Finland and Lausanne, Switzerland. After his graduation in 2002 he received a research and study scholarship from the German Academic Exchange Service DAAD, spending a 10-month research and study leave in 2002 and 2003 at the chair for embedded security at the Ruhr-Universität Bochum, Germany. In 2012 he joined the Fraunhofer Institute for Secure Information Technology (SIT) in Darmstadt, Germany, soon thereafter joining Technische Universität Darmstadt in 2013, where he has been working since as a research assistant at the System Security Lab of the Department of Computer Science. His research interests have been focused on the use of context information for security applications as well as novel security solutions for the Internet of Things (IoT).

Awards

Best Poster / Demo Award, ICDCS 2017

Markus Miettinen, Samuel Marchal, Ibbad Hafeez, Tommaso Frassetto, N. Asokan, Ahmad-Reza Sadeghi and Sasu Tarkoma. "IoT Sentinel Demo: Automated Device-Type Identification for Security Enforcement in IoT". in: *Proc. 37th IEEE International Conference on Distributed Computing Systems (ICDCS 2017)*. Atlanta, GA, USA: IEEE, June 2017. DOI: 10.1109/ICDCS.2017.284

Best Paper Award, ASIACCS 2014

Markus Miettinen, Stephan Heuser, Wiebke Kronz, Ahmad-Reza Sadeghi and N. Asokan. "ConXsense – Context Profiling and Classification for Context-Aware Access Control". In: *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2014)*. ACM. Kyoto, Japan, June 2014. DOI: 10.1145/2590296.2590337

Best Demo Award, PerCom 2011

A. Gupta, M. Miettinen and N. Asokan. "Using context-profiling to aid access control decisions in mobile devices". In: *2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. Mar. 2011, pp. 310–312. DOI: 10.1109/PERCOMW.2011.5766891

Service to the Scientific Community

Programme Committee Memberships

COSDEO2018 6th workshop on Context Systems Design, Evaluation and Optimization

IOTPTS 2017 3rd International Workshop on IoT Privacy, Trust, and Security

SIOT 2017 International Workshop on Secure Internet of Things 2017

IOTPTS 2016 2nd International Workshop on IoT Privacy, Trust, and Security

SIOT 2016 International Workshop on Secure Internet of Things 2016

IW5GS 2016 2nd International Workshop on 5G Security

SIOT 2015 International Workshop on Secure Internet of Things

IW5GS 1st IEEE International Workshop on 5G Security

IOTPTS 2015 Workshop on IoT Privacy, Trust and Security

SMPE 2014 Workshop on Security and Privacy aspects of Mobile Environments

Peer-Reviewed Publications

Markus Miettinen and N. Asokan. "Ad-hoc key agreement: A brief history and the challenges ahead". In: *Computer Communications* 131 (2018). COMCOM 40 years, pp. 32–34. ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2018.07.030. URL: http://www.sciencedirect.com/science/article/pii/S0140366418302007

Markus Miettinen, Thien Duc Nguyen, N. Asokan and Ahmad-Reza Sadeghi. "Revisiting Context-Based Pairing in IoT". in: *Proceedings of the 55th Design Automation Conference (DAC)*. ACM, June 2018. DOI: 10.1145/3195970.3196106

Hossein Fereidooni, Jiska Classen, Tom Spink, Paul Patras, Markus Miettinen, Ahmad-Reza Sadeghi, Matthias Hollick and Mauro Conti. "Breaking Fitness Records Without Moving: Reverse Engineering and Spoofing Fitbit". In: *Research in Attacks, Intrusions, and Defenses*. Ed. by Marc Dacier, Michael Bailey, Michalis Polychronakis and Manos Antonakakis. Springer International Publishing, 2017, pp. 48–69. ISBN: 978-3-319-66332-6

Hossein Fereidooni, Tommaso Frassetto, Markus Miettinen, Ahmad-Reza Sadeghi and Mauro Conti. "Fitness Trackers: Fit for Health but Unfit for Security and Privacy". In: *The Second IEEE International Workshop on Safe, Energy-Aware, & Reliable Connected Health (CHASE-SEARCH)*. Philadelphia, Pennsylvania, USA, July 2017. DOI: 10.1109/CHASE.2017.54

Markus Miettinen, Samuel Marchal, Ibbad Hafeez, N. Asokan, Ahmad-Reza Sadeghi and Sasu Tarkoma. "IoT Sentinel: Automated Device-Type Identification for Security Enforcement in IoT". in: *Proc. 37th IEEE International Conference on Distributed Computing Systems (ICDCS 2017)*. June 2017. DOI: 10.1109/ICDCS.2017.283

M. Lacoste, M. Miettinen, N. Neves, F. M. V. Ramos, M. Vukolic, F. Charmet, R. Yaich, K. Oborzynski, G. Vernekar and P. Sousa. "User-Centric Security and Dependability in the Clouds-of-Clouds". In: *IEEE Cloud Computing* 3.5 (Sept. 2016), pp. 64–75. ISSN: 2325-6095. DOI: 10.1109/MCC.2016.110

Trinh Minh Tri Do, Olivier Dousse, Markus Miettinen and Daniel Gatica-Perez. "A probabilistic kernel method for human mobility prediction with smartphones". In: *Pervasive and Mobile Computing* 20 (2015), pp. 13–28. ISSN: 1574-1192. DOI: http://dx.doi.org/10.1016/j.pmcj.2014.09.001. URL: http://www.sciencedirect.com/science/article/pii/S1574119214001539

Markus Miettinen, N. Asokan, Farinaz Koushanfar, Thien Duc Nguyen, Jon Rios, Ahmad-Reza Sadeghi, Majid Sobhani and Sudha Yellapantula. "I know where you

are: Proofs of Presence resilient to malicious provers". In: *10th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2015)*. Apr. 2015. DOI: 10.1145/2714576.2714634

Markus Miettinen, N. Asokan, Thien Duc Nguyen, Ahmad-Reza Sadeghi and Majid Sobhani. "Context-Based Zero-Interaction Pairing and Key Evolution for Advanced Personal Devices". In: *Proc. ACM Conference on Computer and Communications Security*. Scottsdale, AZ, USA: ACM, Nov. 2014. DOI: 10.1145/2660267.2660334

Markus Miettinen, Stephan Heuser, Wiebke Kronz, Ahmad-Reza Sadeghi and N. Asokan. "ConXsense – Context Profiling and Classification for Context-Aware Access Control". In: *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2014)*. ACM. Kyoto, Japan, June 2014. DOI: 10.1145/2590296.2590337

Juha K. Laurila, Daniel Gatica-Perez, Imad Aad, Jan Blom, Olivier Bornet, Trinh Minh Tri Do, Olivier Dousse, Julien Eberle and Markus Miettinen. "From big smartphone data to worldwide research: The Mobile Data Challenge". In: *Pervasive and Mobile Computing* 9.6 (2013). Mobile Data Challenge, pp. 752–771. ISSN: 1574-1192. DOI: http://dx.doi.org/10.1016/j.pmcj.2013.07.014. URL: http://www.sciencedirect.com/science/article/pii/S1574119213000965

Aditi Gupta, Markus Miettinen, N. Asokan and Marcin Nagy. "Intuitive Security Policy Configuration in Mobile Devices Using Context Profiling". In: *International Conference on Privacy, Security, Risk and Trust (PASSAT), and 2012 International Confernece on Social Computing (SocialCom)*. IEEE, Sept. 2012, pp. 471–480. ISBN: 978-1-4673-5638-1. DOI: 10.1109/SocialCom-PASSAT.2012.60

Yiyun Shen, M. Miettinen, P. Moen and L. Kutvonen. "Privacy Preservation Approach in Service Ecosystems". In: *Enterprise Distributed Object Computing Conference Workshops (EDOCW), 2011 15th IEEE International*. Helsinki, Finland: IEEE, Aug. 2011, pp. 283–292. DOI: 10.1109/EDOCW.2011.59

Markus Miettinen and N. Asokan. "Towards Security Policy Decisions Based on Context Profiling". In: *Proceedings of the 3rd ACM Workshop on Artificial Intelligence and Security*. AISec '10. Chicago, Illinois, USA: ACM, 2010, pp. 19–23. ISBN: 978-1-4503-0088-9. DOI: 10.1145/1866423.1866428

Perttu Halonen, Markus Miettinen and Kimmo Hätönen. "Computer Log Anomaly Detection Using Frequent Episodes". In: *Artificial Intelligence Applications and Innovations III*. ed. by L. Illiadis, I. Vlahavas and M. Bramer. Vol. 296. IFIP Advances in Information and Communication Technology. Boston: Springer, 2009, pp. 417–422. DOI: 10.1007/978-1-4419-0221-4_49. URL: http://dx.doi.org/10.1007/978-1-4419-0221-4_49

Heikki Kokkinen, Mikko V. J. Heikkinen and Markus Miettinen. "Post-Payment Copyright System versus Online Music Shop: Business Model and Privacy". In: *International Journal on Advances in Security* 2.2&3 (2009), pp. 112–128. URL: http://www.iariajournals.org/security/sec_v2_n23_2009_paged.pdf

A. Battestini, C. Del Rosso, A. Flanagan and M. Miettinen. "Creating Next Generation Applications and Services for Mobile Devices: Challenges and Opportunities". In: *Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on*. 2007, pp. 1–4. DOI: 10.1109/PIMRC.2007.4394846. URL: http://dx.doi.org/10.1109/PIMRC.2007.4394846

M. Miettinen, P. Halonen and K. Hätonen. "Host-Based Intrusion Detection for Advanced Mobile Devices". In: *Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA 2006)*. Vol. 2. IEEE Computer Society, Apr. 2006, pp. 72–76. DOI: 10.1109/AINA.2006.192. URL: http://doi.ieeecomputersociety.org/10.1109/AINA.2006.192

Kimmo Hätönen, Mika Klemettinen and Markus Miettinen. "Remarks on the Industrial Application of Inductive Database Technologies". In: *Constraint-Based Mining and Inductive Databases*. Vol. 3848. Lecture Notes in Computer Science. Springer, 2006, pp. 196–215. DOI: 10.1007/11615576_10

Kimmo Hätönen, Jean François Boulicaut, Mika Klemettinen, Markus Miettinen and Cyrille Masson. "Comprehensive Log Compression with Frequent Patterns". In: *Data Warehousing and Knowledge Discovery*. Vol. 2737. Lecture Notes in Computer Science. 10.1007/978-3-540-45228-7_36. Springer Berlin / Heidelberg, 2003, pp. 360–370. URL: http://dx.doi.org/10.1007/978-3-540-45228-7_36

Kimmo Hätönen, Perttu Halonen, Mika Klemettinen and Markus Miettinen. "Queryable lossless log compression". In: *Proceedings of the Second International Workshop on Knowledge Discovery in Inductive Databases, 22 September, Cavtat-Dubrovnik, Croatia*. Ed. by Jean-François Boulicaut and Saso Dzeroski. Rudjer Boskovic Institute, Zagreb, Croatia, 2003, pp. 70–79. ISBN: 953-6690-34-9. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.106.7619&rep=rep1&type=pdf

Book Chapters

Göran Schultz, Olivier Coutand, Ronald van Eijk, Johan Hjelm, Silke Holtmanns, Markus Miettinen and Rinaldo Nani. "Enabling technologies for mobile services : the MobiLife book". In: Wiley, 2007. Chap. Privacy, Trust and Group Communications, pp. 185–225

Technical Reports

Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Minh Hoang Dang, N. Asokan and Ahmad-Reza Sadeghi. "DÏoT: A Crowdsourced Self-learning Approach for Detecting Compromised IoT Devices". In: *CoRR* abs/1804.07474 (2018). arXiv: 1804.07474. URL: http://arxiv.org/abs/1804.07474

A. Acar, H. Fereidooni, T. Abera, A. K. Sikder, M. Miettinen, H. Aksu, M. Conti, A.-R. Sadeghi and A. Selcuk Uluagac. "Peek-a-Boo: I see your smart home activities, even encrypted!" In: *ArXiv e-prints* (Aug. 2018). arXiv: 1808.02741 [cs.CR]

M. Miettinen, P. C. van Oorschot and A.-R. Sadeghi. "Baseline functionality for security and control of commodity IoT devices and domain-controlled device lifecycle management". In: *ArXiv e-prints* (Aug. 2018). arXiv: 1808.03071 [cs.CR]

Posters

Markus Miettinen, Samuel Marchal, Ibbad Hafeez, Tommaso Frassetto, N. Asokan, Ahmad-Reza Sadeghi and Sasu Tarkoma. "IoT Sentinel Demo: Automated Device-Type Identification for Security Enforcement in IoT". in: *Proc. 37th IEEE International Conference on Distributed Computing Systems (ICDCS 2017)*. Atlanta, GA, USA: IEEE, June 2017. DOI: 10.1109/ICDCS.2017.284

Markus Miettinen, Jialin Huang, Thien Duc Nguyen, N. Asokan and Ahmad-Reza Sadeghi. "POSTER: Friend or Foe? Context Authentication for Trust Domain Separation in IoT Environments". In: *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. WiSec '16. Darmstadt, Germany: ACM, 2016, pp. 225–226. ISBN: 978-1-4503-4270-4. DOI: 10.1145/2939918.2942422

Aditi Gupta, Markus Miettinen, Marcin Nagy, N Asokan and Alexandre Wetzel. "PeerSense: Who is near you?" In: *2012 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)2012 IEEE International Conference on Pervasive Computing and Communications Workshops*. IEEE. Lugano,

Switzerland: IEEE, 2012. ISBN: 978-1-4673-0906-6. DOI: 10.1109/PerComW.2012.6197553. URL: http://dx.doi.org/10.1109/PerComW.2012.6197553

A. Gupta, M. Miettinen and N. Asokan. "Using context-profiling to aid access control decisions in mobile devices". In: *2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. Mar. 2011, pp. 310–312. DOI: 10.1109/PERCOMW.2011.5766891

[1] A. Acar, H. Fereidooni, T. Abera, A. K. Sikder, M. Miettinen, H. Aksu, M. Conti, A.-R. Sadeghi and A. Selcuk Uluagac. "Peek-a-Boo: I see your smart home activities, even encrypted!" In: *ArXiv e-prints* (Aug. 2018). arXiv: 1808.02741 [cs.CR].

[2] Agrawal et al. "Fast Discovery of Association Rules." In: *Advances in knowledge discovery and data mining* 12.1 (1996), pp. 307–328.

[3] Manos Antonakakis et al. "Understanding the Mirai Botnet". In: *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, 2017, pp. 1093–1110. ISBN: 978-1-931971-40-9.

[4] Arackaparambil et al. "On the Reliability of Wireless Fingerprinting Using Clock Skews". In: *Proceedings of the Third ACM Conference on Wireless Network Security*. WiSec '10. Hoboken, New Jersey, USA: ACM, 2010, pp. 169–174. ISBN: 978-1-60558-923-7. DOI: 10.1145/1741866.1741894. URL: http://doi.acm.org/10.1145/1741866.1741894.

[5] Guangdong Bai, Liang Gu, Tao Feng, Yao Guo and Xiangqun Chen. "Context-Aware Usage Control for Android". In: *Security and Privacy in Communication Networks*. Ed. by Sushil Jajodia and Jianying Zhou. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 326–343. ISBN: 978-3-642-16161-2.

[6] Dirk Balfanz, Diana K Smetters, Paul Stewart and H Chi Wong. "Talking to strangers: Authentication in ad-hoc wireless networks". In: *Proceedings of the 9th Annual Network and Distributed System Security Symposium (NDSS)* (2002), pp. 7–19.

[7] Boaz Barak, Yevgeniy Dodis, Hugo Krawczyk, Olivier Pereira, Krzysztof Pietrzak, François-Xavier Standaert and Yu Yu. "Leftover Hash Lemma, Revisited". In: *Proc. 31st Annual Cryptology Conference (CRYPTO 2011)*. ISBN: 978-3-642-22792-9.

[8] A. Battestini, C. Del Rosso, A. Flanagan and M. Miettinen. "Creating Next Generation Applications and Services for Mobile Devices: Challenges and Opportunities". In: *Personal, Indoor and Mobile Radio Communications, 2007. PIMRC 2007. IEEE 18th International Symposium on*. 2007, pp. 1–4. DOI: 10.1109/PIMRC.2007.4394846. URL: http://dx.doi.org/10.1109/PIMRC.2007.4394846.

[9] Lujo Bauer, Cristian Bravo-Lillo, Elli Fragkaki and William Melicher. "A comparison of users' perceptions of and willingness to use Google, Facebook, and Google+ single-sign-on functionality". In: *DIM'13, Proceedings of the 2013 ACM Workshop on Digital Identity Management, Berlin, Germany, November 8, 2013*. 2013, pp. 25–36. DOI: 10.1145/2517881.2517886.

[10] S. M. Bellovin and M. Merritt. "Encrypted key exchange: password-based protocols secure against dictionary attacks". In: *Proceedings 1992 IEEE Computer Society Symposium on Research in Security and Privacy*. May 1992, pp. 72–84. DOI: 10.1109/RISP.1992.213269.

[11] Bertini et al. "Profile Resolution Across Multilayer Networks Through Smartphone Camera Fingerprint". In: *Proceedings of the 19th International Database Engineering & Applications Symposium*. IDEAS '15. Yokohama, Japan: ACM, 2014, pp. 23–32. ISBN: 978-1-4503-3414-3. DOI: 10.1145/2790755.2790765. URL: http://doi.acm.org/10.1145/2790755.2790765.

[12] Big Switch Networks. *Project Floodlight - Floodlight OpenFlow Controller*. http://www.projectfloodlight.org/floodlight/. [Accessed: 2016-09-17]. Oct. 2016.

[13] Hristo Bojinov, Yan Michalevsky, Gabi Nakibly and Dan Boneh. "Mobile Device Identification via Sensor Fingerprinting". In: *CoRR* abs/1408.1416 (2014). arXiv: 1408.1416. URL: http://arxiv.org/abs/1408.1416.

[14] Gilles Brassard and Louis Salvail. "Secret-Key Reconciliation by Public Discussion". In: *Proc. Workshop on the Theory and Application of Cryptographic Techniques Lofthus (EUROCRYPT '93), Norway, May 23–27, 1993*. ISBN: 978-3-540-48285-7.

[15] Sergey Bratus, Cory Cornelius, David Kotz and Daniel Peebles. "Active Behavioral Fingerprinting of Wireless Devices". In: *Proceedings of the First ACM Conference on Wireless Network Security*. WiSec '08. Alexandria, VA, USA: ACM, 2008, pp. 56–61. ISBN: 978-1-59593-814-5. DOI: 10.1145/1352533.1352543. URL: http://doi.acm.org/10.1145/1352533.1352543.

[16] Leo Breiman. "Random Forests". In: *Machine Learning* 45.1 (2001), pp. 5–32.

[17] Vladimir Brik, Suman Banerjee, Marco Gruteser and Sangho Oh. "Wireless Device Identification with Radiometric Signatures". In: *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*. MobiCom '08. San Francisco, California, USA: ACM, 2008, pp. 116–127. ISBN: 978-1-60558-096-8. DOI: 10.1145/1409944.1409959. URL: http://doi.acm.org/10.1145/1409944.1409959.

[18] Sven Bugiel, Stephan Heuser and Ahmad-Reza Sadeghi. "Flexible and Fine-Grained Mandatory Access Control on Android for Diverse Security and Privacy Policies". In: *22nd USENIX Security Symposium (USENIX Security '13)*. USENIX, 2013. URL: https://enigma.usenix.org/sites/default/files/sec13_proceedings_interior.pdf#page=139.

[19] *BugTraq Mailing list*. Symantec Corporation. 28th Aug. 2018. URL: https://www.securityfocus.com/archive/1.

[20] Johnny Cache. "Fingerprinting 802.11 Implementations via Statistical Analysis of the Duration Field". In: *Uninformed .org* 5 (2006). URL: http://www.uninformed.org/?v=5&a=1&t=pdf.

[21] Cameron Camp. *The BYOD security challenge: How scary is the iPad, tablet, smartphone surge?* 28th Feb. 2012. URL: https://www.welivesecurity.com/2012/02/28/sizing-up-the-byod-security-challenge/.

[22] Bogdan Carbunar, Radu Sion, Rahul Potharaju and Moussa Ehsan. "The Shy Mayor: Private Badges in GeoSocial Networks". In: *Applied Cryptography and Network Security*. Ed. by Feng Bao, Pierangela Samarati and Jianying Zhou. Vol. 7341. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, pp. 436–454. ISBN: 978-3-642-31283-0. DOI: 10.1007/978-3-642-31284-7_26. URL: http://dx.doi.org/10.1007/978-3-642-31284-7_26.

[23] Haowen Chan, A. Perrig and D. Song. "Random key predistribution schemes for sensor networks". In: *Proc. 2003 IEEE Symposium on Security and Privacy*. May 2003, pp. 197–213. DOI: 10.1109/SECPRI.2003.1199337.

[24] M. Conti, B. Crispo, E. Fernandes and Y. Zhauniarovich. "CRêPE: A System for Enforcing Fine-Grained Context-Related Policies on Android". In: *Information Forensics and Security, IEEE Transactions on* 7.5 (2012), pp. 1426–1438. ISSN: 1556-6013. DOI: 10.1109/TIFS.2012.2204249.

[25] Core Security. *AVTECH DVR multiple vulnerabilities. Security Advisory*. http://www.coresecurity.com/advisories/avtech-dvr-multiple-vulnerabilities. [Accessed: 2016-03-29]. Aug. 2013.

[26] M.J. Covington, P. Fogla, Zhiyuan Zhan and M. Ahamad. "A context-aware security architecture for emerging applications". In: *18th Annual Computer Security Applications Conference*. 2002, pp. 249–258. DOI: 10.1109/CSAC.2002.1176296.

[27] Maria Luisa Damiani, Elisa Bertino, Barbara Catania and Paolo Perlasca. "GEO-RBAC: A spatially aware RBAC". In: *ACM Trans. Inf. Syst. Secur.* 10.1 (Feb. 2007). ISSN: 1094-9224. DOI:

10.1145/1210263.1210265. URL: http://doi.acm.org/10.1145/1210263.1210265.

[28] Ghada Dessouky, Shaza Zeitouni, Thomas Nyman, Andrew Paverd, Lucas Davi, Patrick Koeberl, N Asokan and Ahmad-Reza Sadeghi. "LO-FAT: Low-Overhead Control Flow ATtestation in Hardware". In: *Design Automation Conference (DAC), 2017 54th ACM/EDAC/IEEE*. IEEE. 2017, pp. 1–6.

[29] W. Diffie and M. Hellman. "New directions in cryptography". In: *IEEE Transactions on Information Theory* 22.6 (Nov. 1976), pp. 644–654. ISSN: 0018-9448. DOI: 10.1109/TIT.1976.1055638.

[30] Trinh Minh Tri Do, Olivier Dousse, Markus Miettinen and Daniel Gatica-Perez. "A probabilistic kernel method for human mobility prediction with smartphones". In: *Pervasive and Mobile Computing* 20 (2015), pp. 13–28. ISSN: 1574-1192. DOI: http://dx.doi.org/10.1016/j.pmcj.2014.09.001. URL: http://www.sciencedirect.com/science/article/pii/S1574119214001539.

[31] Yevgeniy Dodis, Leonid Reyzin and Adam Smith. "Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data". In: *Proc. Intl. Conf. on the Theory and Applications of Cryptographic Techniques (EUROCRYPT 2004), Interlaken, Switzerland, May 2-6, 2004*. ISBN: 978-3-540-24676-3.

[32] O. Dousse, J. Eberle and M. Mertens. "Place Learning via Direct WiFi Fingerprint Clustering". In: *IEEE 13th International Conference on Mobile Data Management (MDM)*. 2012, pp. 282–287. DOI: 10.1109/MDM.2012.46.

[33] N. Eagle and A. Pentland. "Social serendipity: mobilizing social software". In: *Pervasive Computing, IEEE* 4.2 (2005), pp. 28–34. ISSN: 1536-1268. DOI: 10.1109/MPRV.2005.37.

[34] S. Edwards and I. Profetis. *Hajime: Analysis of a decentralized internet worm for IoT devices*. Tech. rep. Rapidity Networks, 2016. URL: https://security.rapiditynetworks.com/publications/2016-10-16/hajime.pdf (visited on 16/01/2018).

[35] Laurent Eschenauer and Virgil D. Gligor. "A Key-management Scheme for Distributed Sensor Networks". In: *Proc. 9th ACM Conference on Computer and Communications Security*. CCS '02. Washington, DC, USA: ACM, 2002, pp. 41–47. ISBN: 1-58113-612-9. DOI: 10.1145/586110.586117.

[36] *F-Secure SENSE*. 2016. URL: https://www.f-secure.com/en/web/home_global/sense (visited on 10/09/2018).

[37] Qian Feng, Rundong Zhou, Chengcheng Xu, Yao Cheng, Brian Testa and Heng Yin. "Scalable Graph-based Bug Search for Firmware Images". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. CCS '16. Vienna, Austria: ACM, 2016, pp. 480–491. ISBN: 978-1-4503-4139-4. DOI: 10.1145/2976749.2978370. URL: http://doi.acm.org/10.1145/2976749.2978370.

[38] Hossein Fereidooni, Jiska Classen, Tom Spink, Paul Patras, Markus Miettinen, Ahmad-Reza Sadeghi, Matthias Hollick and Mauro Conti. "Breaking Fitness Records Without Moving: Reverse Engineering and Spoofing Fitbit". In: *Research in Attacks, Intrusions, and Defenses*. Ed. by Marc Dacier, Michael Bailey, Michalis Polychronakis and Manos Antonakakis. Springer International Publishing, 2017, pp. 48–69. ISBN: 978-3-319-66332-6.

[39] Hossein Fereidooni, Tommaso Frassetto, Markus Miettinen, Ahmad-Reza Sadeghi and Mauro Conti. "Fitness Trackers: Fit for Health but Unfit for Security and Privacy". In: *The Second IEEE International Workshop on Safe, Energy-Aware, & Reliable Connected Health (CHASE-SEARCH)*. Philadelphia, Pennsylvania, USA, July 2017. DOI: 10.1109/CHASE.2017.54.

[40] Jason Franklin, Damon McCoy, Parisa Tabriz, Vicentiu Neagoe, Jamie Van Randwyk and Douglas Sicker. "Passive Data Link Layer 802.11 Wireless Device Driver Fingerprinting". In: *Proceedings of the 15th Conference on USENIX Security Symposium*. Vol. 15. USENIX-SS'06. Berkeley, CA, USA: USENIX Association, 2006. URL: http://static.usenix.org/event/sec06/tech/full_papers/franklin/franklin.pdf.

[41] Michael J. Freedman, Kobbi Nissim and Benny Pinkas. "Efficient Private Matching and Set Intersection". In: *Advances in Cryptology - EUROCRYPT 2004*. Ed. by Christian Cachin and Jan L. Camenisch. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 1–19. ISBN: 978-3-540-24676-3.

[42] Ke Gao, C. Corbett and R. Beyah. "A passive approach to wireless device fingerprinting". In: *2010 IEEE/IFIP International Conference on Dependable Systems Networks (DSN)*. June 2010, pp. 383–392. DOI: 10.1109/DSN.2010.5544294.

[43] Inc. Gartner. *Gartner Says 8.4 Billion Connected "Things" Will Be in Use in 2017, Up 31 Percent From 2016*. 7th Feb. 2017. URL: https://www.gartner.com/en/newsroom/press-releases/2017-02-07-gartner-says-8-billion-connected-things-will-be-in-use-in-2017-up-31-percent-from-2016 (visited on 05/10/2018).

[44] Stylianos Gisdakis, Thanassis Giannetsos and Panos Papadimitratos. "SHIELD: A Data Verification Framework for Participatory Sensing Systems". In: *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. WiSec '15. New York, New York: ACM, 2015, 16:1–16:12. ISBN: 978-1-4503-3623-9. DOI: 10.1145/2766498.2766503. URL: http://doi.acm.org/10.1145/2766498.2766503.

[45] John Greenough. *How the 'Internet of Things' will impact consumers, businesses, and governments in 2016 and beyond*. Business Insider. URL: http://www.businessinsider.de/how-the-internet-of-things-market-will-grow-2014-10 (visited on 19/09/2016).

[46] A. Gupta, M. Miettinen and N. Asokan. "Using context-profiling to aid access control decisions in mobile devices". In: *2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*. Mar. 2011, pp. 310–312. DOI: 10.1109/PERCOMW.2011.5766891.

[47] Aditi Gupta, Markus Miettinen, N. Asokan and Marcin Nagy. "Intuitive Security Policy Configuration in Mobile Devices Using Context Profiling". In: *International Conference on Privacy, Security, Risk and Trust (PASSAT), and 2012 International Confernece on Social Computing (SocialCom)*. IEEE, Sept. 2012, pp. 471–480. ISBN: 978-1-4673-5638-1. DOI: 10.1109/SocialCom-PASSAT.2012.60.

[48] Aditi Gupta, Markus Miettinen, Marcin Nagy, N Asokan and Alexandre Wetzel. "PeerSense: Who is near you?" In: *2012 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)2012 IEEE International Conference on Pervasive Computing and Communications Workshops*. IEEE. Lugano, Switzerland: IEEE, 2012. ISBN: 978-1-4673-0906-6. DOI: 10.1109/PerComW.2012.6197553. URL: http://dx.doi.org/10.1109/PerComW.2012.6197553.

[49] Tzipora Halevi, Di Ma, Nitesh Saxena and Tuo Xiang. "Secure Proximity Detection for NFC Devices Based on Ambient Sensor Data". In: *Computer Security ESORICS 2012*. Ed. by Sara Foresti, Moti Yung and Fabio Martinelli. Vol. 7459. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, pp. 379–396. ISBN: 978-3-642-33166-4. DOI: 10.1007/978-3-642-33167-1_22.

[50] Hall et al. "The WEKA data mining software: an update". In: *SIGKDD Explor. Newsl.* 11.1 (Nov. 2009), pp. 10–18. ISSN: 1931-0145. DOI: 10.1145/1656274.1656278. URL: http://doi.acm.org/10.1145/1656274.1656278.

[51] Roger Hallman, Josiah Bryan, Geancarlo Palavicini, Joseph Divita and Jose Romero-Mariona. "IoDDoS—The Internet of Distributed Denial of Sevice Attacks-A Case Study of the Mirai Malware and IoT Based Botnets". In: *Proceedings of the 2nd International Conference on Internet of Things, Big Data and Security (IoTBDS 2017)*. 2017, pp. 47–58.

[52] Perttu Halonen, Markus Miettinen and Kimmo Hätönen. "Computer Log Anomaly Detection Using Frequent Episodes". In: *Artificial Intelligence Applications and Innovations III*. Ed. by L. Iliadis, I. Vlahavas and M. Bramer. Vol. 296. IFIP Advances in Information and Communication Technology. Boston: Springer, 2009, pp. 417–422. DOI: 10.1007/978-1-4419-0221-4_49. URL: http://dx.doi.org/10.1007/978-1-4419-0221-4_49.

[53] Eiji Hayashi, Sauvik Das, Shahriyar Amini, Jason Hong and Ian Oakley. "CASA: context-aware scalable authentication". In: *Ninth Symposium on Usable Privacy and Security*. SOUPS '13. Newcastle, United Kingdom: ACM, 2013, 3:1–3:10. ISBN: 978-1-4503-2319-2. DOI: 10.1145/2501604.2501607. URL: http://doi.acm.org/10.1145/2501604.2501607.

[54] J. L. Hernández-Ramos, M. P. Pawlowski, A. J. Jara, A. F. Skarmeta and L. Ladid. "Toward a Lightweight Authentication and Authorization Framework for Smart Objects". In: *IEEE Journal on Selected Areas in Communications* 33.4 (2015), pp. 690–702.

[55] Yih-Chun Hu, A. Perrig and D.B. Johnson. "Packet leashes: a defense against wormhole attacks in wireless networks". In: *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*. Vol. 3. 2003, 1976–1986 vol.3. DOI: 10.1109/INFCOM.2003.1209219.

[56] Hull et al. "Enabling context-aware and privacy-conscious user data sharing". In: *2004 IEEE International Conference on Mobile Data Management*. 2004, pp. 187–198. DOI: 10.1109/MDM.2004.1263065.

[57] Kimmo Hätönen, Jean François Boulicaut, Mika Klemettinen, Markus Miettinen and Cyrille Masson. "Comprehensive Log Compression with Frequent Patterns". In: *Data Warehousing and Knowledge Discovery*. Vol. 2737. Lecture Notes in Computer Science. 10.1007/978-3-540-45228-7_36. Springer Berlin / Heidelberg, 2003, pp. 360–370. URL: http://dx.doi.org/10.1007/978-3-540-45228-7_36.

[58] Kimmo Hätönen, Perttu Halonen, Mika Klemettinen and Markus Miettinen. "Queryable lossless log compression". In: *Proceedings of the Second International Workshop on Knowledge Discovery in Inductive Databases, 22 September, Cavtat-Dubrovnik, Croatia*. Ed. by Jean-François Boulicaut and Saso Dzeroski. Rudjer

Boskovic Institute, Zagreb, Croatia, 2003, pp. 70–79. ISBN: 953-6690-34-9. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.106.7619&rep=rep1&type=pdf.

[59] Kimmo Hätönen, Mika Klemettinen and Markus Miettinen. "Remarks on the Industrial Application of Inductive Database Technologies". In: *Constraint-Based Mining and Inductive Databases*. Vol. 3848. Lecture Notes in Computer Science. Springer, 2006, pp. 196–215. DOI: 10.1007/11615576_10.

[60] *IEEE 802.11i-2004 standard amendment 6: Medium access control (MAC) security enhancements*. IEEE. 23rd July 2004.

[61] ECMA International. *ECMA-404: The JSON data interchange syntax*. 2nd ed. ECMA Standard. Dec. 2017. URL: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf.

[62] S. Jana and S. K. Kasera. "On Fast and Accurate Detection of Unauthorized Wireless Access Points Using Clock Skews". In: *IEEE Transactions on Mobile Computing* 9.3 (Mar. 2010), pp. 449–462. ISSN: 1536-1233. DOI: 10.1109/TMC.2009.145.

[63] Suman Jana, Sriram Nandha Premnath, Mike Clark, Sneha K. Kasera, Neal Patwari and Srikanth V. Krishnamurthy. "On the Effectiveness of Secret Key Extraction from Wireless Signal Strength in Real Environments". In: *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking*. MobiCom '09. Beijing, China: ACM, 2009, pp. 321–332. ISBN: 978-1-60558-702-8. DOI: 10.1145/1614320.1614356.

[64] Ari Juels and Madhu Sudan. "A Fuzzy Vault Scheme". English. In: *Designs, Codes and Cryptography* 38.2 (2006), pp. 237–257. ISSN: 0925-1022. DOI: 10.1007/s10623-005-6343-z. URL: http://dx.doi.org/10.1007/s10623-005-6343-z.

[65] Andre Kalamandeen, Adin Scannell, Eyal de Lara, Anmol Sheth and Anthony LaMarca. "Ensemble: Cooperative Proximity-based Authentication". In: *Proc. 8th Intl. Conf. on Mobile Systems, Applications, and Services (MobiSys '10), 2010*. San Francisco, California, USA, 2010, pp. 331–344. ISBN: 978-1-60558-985-5. DOI: 10.1145/1814433.1814466.

[66] *Kali Linux - Penetration testing and ethical hacking Linux distribution*. 2018. URL: https://www.kali.org/ (visited on 29/08/2018).

[67] Jong Hee Kang, William Welbourne, Benjamin Stewart and Gaetano Borriello. "Extracting places from traces of locations". In: *SIGMOBILE Mob. Comput. Commun. Rev.* 9.3 (July 2005), pp. 58–68. ISSN: 1559-1662. DOI: 10.1145/1094549.1094558. URL: http://doi.acm.org/10.1145/1094549.1094558.

[68] Patrick Gage Kelley, Paul Hankes Drielsma, Norman M. Sadeh and Lorrie Faith Cranor. "User-controllable learning of security and privacy policies". In: *1st ACM Workshop on Workshop on AISec*. AISec '08. New York, NY, USA: ACM, 2008, pp. 11–18. DOI: 10.1145/1456377.1456380. URL: http://doi.acm.org/10.1145/1456377.1456380.

[69] Z. Kfir and A. Wool. "Picking Virtual Pockets using Relay Attacks on Contactless Smartcard". In: *First International Conference on Security and Privacy for Emerging Areas in Communications Networks (SECURECOMM'05)*. Sept. 2005, pp. 47–58. DOI: 10.1109/SECURECOMM.2005.32.

[70] T. Kohno, A. Broido and K. C. Claffy. "Remote physical device fingerprinting". In: *IEEE Transactions on Dependable and Secure Computing* 2.2 (Apr. 2005), pp. 93–108. ISSN: 1545-5971. DOI: 10.1109/TDSC.2005.26.

[71] Heikki Kokkinen, Mikko V. J. Heikkinen and Markus Miettinen. "Post-Payment Copyright System versus Online Music Shop: Business Model and Privacy". In: *International Journal on Advances in Security* 2.2&3 (2009), pp. 112–128. URL: http://www.iariajournals.org/security/sec_v2_n23_2009_paged.pdf.

[72] Andreas Kurtz, Hugo Gascon, Tobias Becker, Konrad Rieck and Felix Freiling. "Fingerprinting Mobile Devices Using Personalized Configurations". In: *Proceedings on Privacy Enhancing Technologies* 2016.1 (2016), pp. 4–19. URL: https://content.sciendo.com/view/journals/popets/2016/1/article-p4.xml.

[73] M. Lacoste, M. Miettinen, N. Neves, F. M. V. Ramos, M. Vukolic, F. Charmet, R. Yaich, K. Oborzynski, G. Vernekar and P. Sousa. "User-Centric Security and Dependability in the Clouds-of-Clouds". In: *IEEE Cloud Computing* 3.5 (Sept. 2016), pp. 64–75. ISSN: 2325-6095. DOI: 10.1109/MCC.2016.110.

[74] Juha K. Laurila, Daniel Gatica-Perez, Imad Aad, Jan Blom, Olivier Bornet, Trinh Minh Tri Do, Olivier Dousse, Julien Eberle and Markus Miettinen. "From big smartphone data to worldwide research: The Mobile Data Challenge". In: *Pervasive and Mobile Computing* 9.6 (2013). Mobile Data Challenge, pp. 752–771. ISSN: 1574-1192. DOI: http://dx.doi.org/10.1016/j.pmcj.2013.07.014. URL: http://www.sciencedirect.com/science/article/pii/S1574119213000965.

[75] Chieh-Jan Mike Liang, Börje F. Karlsson, Nicholas D. Lane, Feng Zhao, Junbei Zhang, Zheyi Pan, Zhao Li and Yong Yu. "SIFT: Building an Internet of Safe Things". In: *Proceedings of the 14th International Conference on Information Processing in Sensor Networks*. IPSN '15. Seattle, Washington: ACM, 2015,

pp. 298–309. ISBN: 978-1-4503-3475-4. DOI: 10.1145/2737095.2737115. URL: http://doi.acm.org/10.1145/2737095.2737115.

[76] Donggang Liu, Peng Ning and Rongfang Li. "Establishing Pairwise Keys in Distributed Sensor Networks". In: *ACM Trans. Inf. Syst. Secur.* 8.1 (Feb. 2005), pp. 41–77. ISSN: 1094-9224. DOI: 10.1145/1053283.1053287.

[77] David Lodge. *Hacking a Wi-Fi Coffee Machine – Part 1*. 2015. URL: https://www.pentestpartners.com/blog/hacking-a-wi-fi-coffee-machine-part-1/ (visited on 29/03/2016).

[78] Anmol Madan, Manuel Cebrian, David Lazer and Alex Pentland. "Social Sensing for Epidemiological Behavior Change". In: *12th ACM International Conference on Ubiquitous Computing*. Ubicomp '10. Copenhagen, Denmark: ACM, 2010, pp. 291–300. ISBN: 978-1-60558-843-8. DOI: 10.1145/1864349.1864394. URL: http://doi.acm.org/10.1145/1864349.1864394.

[79] Jouni Malinen. *hostapd: IEEE 802.11 AP*. 12th Jan. 2013. URL: http://w1.fi/hostapd/ (visited on 29/08/2018).

[80] Justin Manweiler, Ryan Scudellari and Landon P. Cox. "SMILE: Encounter-based Trust for Mobile Social Services". In: *Proceedings of the 16th ACM Conference on Computer and Communications Security*. CCS '09. Chicago, Illinois, USA: ACM, 2009, pp. 246–255. ISBN: 978-1-60558-894-0. DOI: 10.1145/1653662.1653692.

[81] Philip Marquardt, Arunabh Verma, Henry Carter and Patrick Traynor. "(Sp)iPhone: Decoding Vibrations from Nearby Keyboards Using Mobile Phone Accelerometers". In: *18th ACM Conference on Computer and Communications Security*. CCS '11. Chicago, Illinois, USA: ACM, 2011, pp. 551–562. ISBN: 978-1-4503-0948-6. DOI: 10.1145/2046707.2046771. URL: http://doi.acm.org/10.1145/2046707.2046771.

[82] Mathur et al. "ProxiMate: Proximity-based Secure Pairing Using Ambient Wireless Signals". In: *Proc. 9th Intl. Conf. on Mobile Systems, Applications, and Services (MobiSys '11), 2011*. Bethesda, Maryland, USA, pp. 211–224. ISBN: 978-1-4503-0643-0. DOI: 10.1145/1999995.2000016.

[83] C. Maurice, S. Onno, C. Neumann, O. Heen and A. Francillon. "Improving 802.11 fingerprinting of similar devices by cooperative fingerprinting". In: *Proceedings of the 2013 International Conference on Security and Cryptography (SECRYPT)*. 2013, pp. 1–8.

[84] M. Miettinen, P. Halonen and K. Hätonen. "Host-Based Intrusion Detection for Advanced Mobile Devices". In: *Proceedings of the 20th International Conference on Advanced Information Networking and Applications (AINA 2006)*. Vol. 2. IEEE Computer

Society, Apr. 2006, pp. 72–76. DOI: 10.1109/AINA.2006.192. URL: http://doi.ieeecomputersociety.org/10.1109/AINA.2006.192.

[85] M. Miettinen, P. C. van Oorschot and A.-R. Sadeghi. "Baseline functionality for security and control of commodity IoT devices and domain-controlled device lifecycle management". In: *ArXiv e-prints* (Aug. 2018). arXiv: 1808.03071 [cs.CR].

[86] Markus Miettinen and N. Asokan. "Towards Security Policy Decisions Based on Context Profiling". In: *Proceedings of the 3rd ACM Workshop on Artificial Intelligence and Security*. AISec '10. Chicago, Illinois, USA: ACM, 2010, pp. 19–23. ISBN: 978-1-4503-0088-9. DOI: 10.1145/1866423.1866428.

[87] Markus Miettinen and N. Asokan. "Ad-hoc key agreement: A brief history and the challenges ahead". In: *Computer Communications* 131 (2018). COMCOM 40 years, pp. 32–34. ISSN: 0140-3664. DOI: https://doi.org/10.1016/j.comcom.2018.07.030. URL: http://www.sciencedirect.com/science/article/pii/S0140366418302007.

[88] Markus Miettinen, N. Asokan, Farinaz Koushanfar, Thien Duc Nguyen, Jon Rios, Ahmad-Reza Sadeghi, Majid Sobhani and Sudha Yellapantula. "I know where you are: Proofs of Presence resilient to malicious provers". In: *10th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2015)*. Apr. 2015. DOI: 10.1145/2714576.2714634.

[89] Markus Miettinen, N. Asokan, Thien Duc Nguyen, Ahmad-Reza Sadeghi and Majid Sobhani. "Context-Based Zero-Interaction Pairing and Key Evolution for Advanced Personal Devices". In: *Proc. ACM Conference on Computer and Communications Security*. Scottsdale, AZ, USA: ACM, Nov. 2014. DOI: 10.1145/2660267.2660334.

[90] Markus Miettinen, Stephan Heuser, Wiebke Kronz, Ahmad-Reza Sadeghi and N. Asokan. "ConXsense – Context Profiling and Classification for Context-Aware Access Control". In: *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security (ASIACCS 2014)*. ACM. Kyoto, Japan, June 2014. DOI: 10.1145/2590296.2590337.

[91] Markus Miettinen, Jialin Huang, Thien Duc Nguyen, N. Asokan and Ahmad-Reza Sadeghi. "POSTER: Friend or Foe? Context Authentication for Trust Domain Separation in IoT Environments". In: *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. WiSec '16. Darmstadt, Germany: ACM, 2016, pp. 225–226. ISBN: 978-1-4503-4270-4. DOI: 10.1145/2939918.2942422.

[92] Markus Miettinen, Samuel Marchal, Ibbad Hafeez, N. Asokan, Ahmad-Reza Sadeghi and Sasu Tarkoma. "IoT Sentinel: Automated Device-Type Identification for Security Enforcement in IoT". In: *Proc. 37th IEEE International Conference on Distributed Computing Systems (ICDCS 2017)*. June 2017. DOI: 10.1109/ICDCS.2017.283.

[93] Markus Miettinen, Samuel Marchal, Ibbad Hafeez, Tommaso Frassetto, N. Asokan, Ahmad-Reza Sadeghi and Sasu Tarkoma. "IoT Sentinel Demo: Automated Device-Type Identification for Security Enforcement in IoT". In: *Proc. 37th IEEE International Conference on Distributed Computing Systems (ICDCS 2017)*. Atlanta, GA, USA: IEEE, June 2017. DOI: 10.1109/ICDCS.2017.284.

[94] Markus Miettinen, Thien Duc Nguyen, N. Asokan and Ahmad-Reza Sadeghi. "Revisiting Context-Based Pairing in IoT". In: *Proceedings of the 55th Design Automation Conference (DAC)*. ACM, June 2018. DOI: 10.1145/3195970.3196106.

[95] MITRE Corporation. *Common Vulnerabilities and Exposures*. URL: https://cve.mitre.org/data/downloads/index.html (visited on 27/09/2016).

[96] Raúl Montoliu, Jan Blom and Daniel Gatica-Perez. "Discovering places of interest in everyday life from smartphone data". In: *Multimedia Tools Appl.* 62.1 (2013), pp. 179–207. DOI: 10.1007/s11042-011-0982-z.

[97] V. Mora-Afonso, P. Caballero-Gil and J. Molina-Gil. "Strong authentication on smart wireless devices". In: *Second International Conference on Future Generation Communication Technologies (FGCT 2013)*. Nov. 2013, pp. 137–142. DOI: 10.1109/FGCT.2013.6767206.

[98] M. J. Moyer and M. Abamad. "Generalized role-based access control". In: *Proceedings 21st International Conference on Distributed Computing Systems*. Apr. 2001, pp. 391–398. DOI: 10.1109/ICDSC.2001.918969.

[99] F.M. Naini, O. Dousse, P. Thiran and M. Vetterli. "Population size estimation using a few individuals as agents". In: *2011 IEEE International Symposium on Information Theory Proceedings (ISIT)*. July 2011, pp. 2499–2503. DOI: 10.1109/ISIT.2011.6034016.

[100] Arvind Narayanan, Narendran Thiagarajan, Mugdha Lakhani, Michael Hamburg and Dan Boneh. "Location Privacy via Private Proximity Testing". In: *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011*. 2011. URL:

http://www.isoc.org/isoc/conferences/ndss/11/pdf/1_3.pdf.

[101] *National Vulnerability Database*. 28th Aug. 2018. URL: http://nvd.nist.gov/.

[102] Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, Minh Hoang Dang, N. Asokan and Ahmad-Reza Sadeghi. "DÏoT: A Crowdsourced Self-learning Approach for Detecting Compromised IoT Devices". In: *CoRR* abs/1804.07474 (2018). arXiv: 1804.07474. URL: http://arxiv.org/abs/1804.07474.

[103] Machigar Ongtang, Stephen McLaughlin, William Enck and Patrick McDaniel. "Semantically rich application-centric security in Android". In: *Security and Communication Networks* 5.6 (), pp. 658–673. DOI: 10.1002/sec.360. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/sec.360. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/sec.360.

[104] Pang et al. "802.11 User Fingerprinting". In: *Proceedings of the 13th Annual ACM International Conference on Mobile Computing and Networking*. MobiCom '07. Montréal, Québec, Canada: ACM, 2007, pp. 99–110. ISBN: 978-1-59593-681-3. DOI: 10.1145/1287853.1287866. URL: http://doi.acm.org/10.1145/1287853.1287866.

[105] Jaehong Park and Ravi Sandhu. "Towards Usage Control Models: Beyond Traditional Access Control". In: *Proceedings of the Seventh ACM Symposium on Access Control Models and Technologies*. SACMAT '02. Monterey, California, USA: ACM, 2002, pp. 57–64. ISBN: 1-58113-496-7. DOI: 10.1145/507711.507722. URL: http://doi.acm.org/10.1145/507711.507722.

[106] Jaehong Park and Ravi Sandhu. "The UCONABC Usage Control Model". In: *ACM Trans. Inf. Syst. Secur.* 7.1 (Feb. 2004), pp. 128–174. ISSN: 1094-9224. DOI: 10.1145/984334.984339. URL: http://doi.acm.org/10.1145/984334.984339.

[107] Ben Pfaff et al. "The Design and Implementation of Open vSwitch". In: *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation*. USENIX Association, 2015, pp. 117–130.

[108] Iasonas Polakis, Stamatis Volanis, Elias Athanasopoulos and Evangelos P. Markatos. "The Man Who Was There: Validating Check-ins in Location-based Services". In: *Proceedings of the 29th Annual Computer Security Applications Conference*. ACSAC '13. New Orleans, Louisiana: ACM, 2013, pp. 19–28. ISBN: 978-1-4503-2015-3. DOI: 10.1145/2523649.2523653. URL: http://doi.acm.org/10.1145/2523649.2523653.

[109] Di Qiu, Dan Boneh, Sherman Lo and Per Enge. "Robust location tag generation from noisy location data for security applications". In: *The Institute of Navigation International Technical Meeting*. 2009, pp. 586–597.

[110] S. V. Radhakrishnan, A. S. Uluagac and R. Beyah. "GTID: A Technique for Physical DeviceandDevice Type Fingerprinting". In: *IEEE Transactions on Dependable and Secure Computing* 12.5 (Sept. 2015), pp. 519–532. ISSN: 1545-5971. DOI: 10.1109/TDSC.2014.2369033.

[111] Radware. *BrickerBot Results In PDoS Attack*. https://security.radware.com/ddos-threats-attacks/brickerbot-pdos-permanent-denial-of-service/. 5th Apr. 2017. (Visited on 16/01/2018).

[112] *Raspberry Pi 2 Model B*. URL: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/ (visited on 31/08/2018).

[113] Shahid Raza, Linus Wallgren and Thiemo Voigt. "SVELTE: Real-time intrusion detection in the Internet of Things". In: *Ad Hoc Networks* 11.8 (2013), pp. 2661–2674. ISSN: 1570-8705. DOI: https://doi.org/10.1016/j.adhoc.2013.04.014. URL: http://www.sciencedirect.com/science/article/pii/S1570870513001005.

[114] I. Reed and G. Solomon. "Polynomial Codes Over Certain Finite Fields". In: *Journal of the Society for Industrial and Applied Mathematics* 8.2 (1960), pp. 300–304. DOI: 10.1137/0108018. eprint: http://epubs.siam.org/doi/pdf/10.1137/0108018. URL: http://epubs.siam.org/doi/abs/10.1137/0108018.

[115] Riva et al. "Progressive authentication: deciding when to authenticate on mobile phones". In: *21st USENIX Security Symposium*. 2012. URL: https://www.usenix.org/system/files/conference/usenixsecurity12/sec12-final154.pdf.

[116] Benjamin I.P. Rubinstein, Blaine Nelson, Ling Huang, Anthony D. Joseph, Shing-hon Lau, Satish Rao, Nina Taft and J. D. Tygar. "ANTIDOTE: Understanding and Defending Against Poisoning of Anomaly Detectors". In: *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*. IMC '09. Chicago, Illinois, USA: ACM, 2009, pp. 1–14. ISBN: 978-1-60558-771-4. DOI: 10.1145/1644893.1644895. URL: http://doi.acm.org/10.1145/1644893.1644895.

[117] Giovanni Russello, Mauro Conti, Bruno Crispo and Earlence Fernandes. "MOSES: supporting operation modes on smartphones". In: *17th ACM symposium on Access Control Models and Technologies*. SACMAT '12. Newark, New Jersey, USA: ACM, 2012, pp. 3–12. ISBN: 978-1-4503-1295-0. DOI: 10.1145/2295136.2295140. URL: http://doi.acm.org/10.1145/2295136.2295140.

[118] Norman Sadeh, Jason Hong, Lorrie Cranor, Ian Fette, Patrick Kelley, Madhu Prabaker and Jinghai Rao. "Understanding and capturing people's privacy policies in a mobile social networking application". English. In: *Personal and Ubiquitous Computing* 13 (6 2009), pp. 401–412. ISSN: 1617-4909. DOI: 10.1007/s00779-008-0214-3. URL: http://dx.doi.org/10.1007/s00779-008-0214-3.

[119] Mehran Sahami, Susan Dumais, David Heckerman and Eric Horvitz. "A Bayesian approach to filtering junk e-mail". In: *Learning for Text Categorization: Papers from the 1998 workshop*. Vol. 62. 1998, pp. 98–105.

[120] R. S. Sandhu, E. J. Coyne, H. L. Feinstein and C. E. Youman. "Role-based access control models". In: *Computer* 29.2 (Feb. 1996), pp. 38–47. ISSN: 0018-9162. DOI: 10.1109/2.485845.

[121] Stefan Saroiu and Alec Wolman. "Enabling New Mobile Applications with Location Proofs". In: *Proceedings of the 10th Workshop on Mobile Computing Systems and Applications*. HotMobile '09. Santa Cruz, California: ACM, 2009, 3:1–3:6. ISBN: 978-1-60558-283-2. DOI: 10.1145/1514411.1514414. URL: http://doi.acm.org/10.1145/1514411.1514414.

[122] Roman Schlegel, Kehuan Zhang, Xiao-yong Zhou, Mehool Intwala, Apu Kapadia and XiaoFeng Wang. "Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones". In: *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011*. 2011. URL: http://www.isoc.org/isoc/conferences/ndss/11/pdf/1_1.pdf.

[123] Göran Schultz, Olivier Coutand, Ronald van Eijk, Johan Hjelm, Silke Holtmanns, Markus Miettinen and Rinaldo Nani. "Enabling technologies for mobile services : the MobiLife book". In: Wiley, 2007. Chap. Privacy, Trust and Group Communications, pp. 185–225.

[124] D. Schürmann and S. Sigg. "Secure Communication Based on Ambient Audio". In: *Mobile Computing, IEEE Transactions on* 12.2 (Feb. 2013), pp. 358–370. ISSN: 1536-1233. DOI: 10.1109/TMC.2011.271.

[125] Scikit Learn. *scikit-learn. Machine Learning in Python*. http://scikit-learn.org.

[126] SEC Consult. *House of Keys: 9 Months later... 40% Worse*. Sept. 2016. URL: http://blog.sec-consult.com/2016/09/house-of-keys-9-months-later-40-worse.html (visited on 07/09/2016).

[127] Senrio. *400,000 Publicly Available IoT Devices Vulnerable to Single Flaw*. [Accessed: 2016-07-07]. URL: http://blog.senr.io/blog/400000-publicly-available-iot-devices-vulnerable-to-single-flaw (visited on 26/09/2016).

[128] Y. Sharaf-Dabbagh and W. Saad. "On the authentication of devices in the Internet of things". In: *2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. June 2016, pp. 1–3. DOI: 10.1109/WoWMoM.2016.7523532.

[129] Shiqi Shen, Shruti Tople and Prateek Saxena. "Auror: Defending Against Poisoning Attacks in Collaborative Deep Learning Systems". In: *Proceedings of the 32Nd Annual Conference on Computer Security Applications*. ACSAC '16. Los Angeles, California, USA: ACM, 2016, pp. 508–519. ISBN: 978-1-4503-4771-6. DOI: 10.1145/2991079.2991125. URL: http://doi.acm.org/10.1145/2991079.2991125.

[130] Yiyun Shen, M. Miettinen, P. Moen and L. Kutvonen. "Privacy Preservation Approach in Service Ecosystems". In: *Enterprise Distributed Object Computing Conference Workshops (EDOCW), 2011 15th IEEE International*. Helsinki, Finland: IEEE, Aug. 2011, pp. 283–292. DOI: 10.1109/EDOCW.2011.59.

[131] Babins Shrestha, Nitesh Saxena, Hien Thi Thu Truong and N. Asokan. "Drone to the Rescue: Relay-Resilient Authentication using Ambient Multi-sensing". In: *Financial Cryptography and Data Security*. Ed. by Nicolas Christin and Reihaneh Safavi-Naini. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 349–364. ISBN: 978-3-662-45472-5.

[132] Robert Siciliano. *More Than 30% of People Don't Password Protect Their Mobile Devices*. 24th Feb. 2013. URL: http://blogs.mcafee.com/consumer/unprotected-mobile-devices.

[133] Bluetooth SIG. *Bluetooth V4.0 Core specification*. URL: https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=229737 (visited on 30/08/2018).

[134] Sivaraman et al. "Smart-Phones Attacking Smart-Homes". In: *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. WiSec '16. Darmstadt, Germany: ACM, 2016, pp. 195–200. ISBN: 978-1-4503-4270-4. DOI: 10.1145/2939918.2939925.

[135] Frank Stajano. "The resurrecting duckling". In: *International workshop on security protocols*. Springer. 1999, pp. 183–194.

[136] StatCounter. *Mobile and tablet internet usage exceeds desktop for first time worldwide*. 1st Nov. 2016. URL: http://gs.statcounter.com/press/mobile-and-tablet-internet-usage-exceeds-desktop-for-first-time-worldwide.

[137] Tcpdump/Libpcap. *TCPDUMP/LIBPCAP public repository*. URL: http://www.tcpdump.org/ (visited on 29/08/2018).

[138] Robert Templeman, Zahid Rahman, David J. Crandall and Apu Kapadia. "PlaceRaider: Virtual Theft in Physical Spaces with Smartphones". In: *20th Annual Network and Distributed System Security Symposium, NDSS 2013, San Diego, California, USA, February 24-27, 2013*. 2013. URL: https://www.ndss-symposium.org/ndss2013/placeraider-virtual-theft-physical-spaces-smartphones.

[139] P. Traynor, R. Kumar, Heesook Choi, Guohong Cao, Sencun Zhu and T. La Porta. "Efficient Hybrid Security Mechanisms for Heterogeneous Sensor Networks". In: *IEEE Transactions on Mobile Computing* 6.6 (June 2007), pp. 663–677. ISSN: 1536-1233. DOI: 10.1109/TMC.2007.1020.

[140] H. T. T. Truong, Xiang Gao, B. Shrestha, N. Saxena, N. Asokan and P. Nurmi. "Comparing and fusing different sensor modalities for relay attack resistance in Zero-Interaction Authentication". In: *2014 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. Mar. 2014, pp. 163–171. DOI: 10.1109/PerCom.2014.6813957.

[141] O. Ureten and N. Serinken. "Wireless security through RF fingerprinting". In: *Canadian Journal of Electrical and Computer Engineering* 32.1 (Winter 2007), pp. 27–33. ISSN: 0840-8688. DOI: 10.1109/CJECE.2007.364330.

[142] Tom Van Goethem, Wout Scheepers, Davy Preuveneers and Wouter Joosen. "Accelerometer-Based Device Fingerprinting for Multi-factor Mobile Authentication". In: *Engineering Secure Software and Systems*. Ed. by Juan Caballero, Eric Bodden and Elias Athanasopoulos. Cham: Springer International Publishing, 2016, pp. 106–121. ISBN: 978-3-319-30806-7.

[143] Alex Varshavsky, Adin Scannell, Anthony LaMarca and Eyal Lara. "Amigo: Proximity-Based Authentication of Mobile Devices". In: *UbiComp 2007: Ubiquitous Computing*. Ed. by John Krumm, GregoryD. Abowd, Aruna Seneviratne and Thomas Strang. Vol. 4717. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, pp. 253–270. ISBN: 978-3-540-74852-6. DOI: 10.1007/978-3-540-74853-3_15.

[144] *What are the current protection features for F-Secure Sense?* URL: https://community.f-secure.com/t5/F-Secure-SENSE/What-are-the-current-protection/ta-p/82972 (visited on 10/09/2018).

[145] Nan Xu, Fan Zhang, Yisha Luo, Weijia Jia, Dong Xuan and Jin Teng. "Stealthy Video Capturer: A New Video-based Spyware in 3G Smartphones". In: *Second ACM Conference on Wireless*

Network Security. WiSec '09. Zurich, Switzerland: ACM, 2009, pp. 69–78. ISBN: 978-1-60558-460-7. DOI: 10.1145/1514274.1514285. URL: http://doi.acm.org/10.1145/1514274.1514285.

[146] T. Yeh, D. Chiu and K. Lu. *Persirai: New Internet of Things (IoT) Botnet Targets IP Cameras*. https://blog.trendmicro.com/trendlabs-security-intelligence/persirai-new-internet-things-iot-botnet-targets-ip-cameras/. TrendMicro.

[147] *Z-Wave Alliance*. 2018. URL: https://z-wavealliance.org/.

[148] Zenger et al. "Authenticated key establishment for low-resource devices exploiting correlated random channels". In: *Computer Networks* 109 (2016). Special issue on Recent Advances in Physical-Layer Security, pp. 105–123. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2016.06.013.

[149] Zheng et al. "Collaborative location and activity recommendations with GPS history data". In: *19th International Conference on World Wide Web*. Ed. by Michael Rappa, Paul Jones, Juliana Freire and Soumen Chakrabarti. New York, NY, USA: ACM, 2010, pp. 1029–1038. ISBN: 978-1-60558-799-8. DOI: 10.1145/1772690.1772795.

[150] *Zigbee Alliance*. 2018. URL: https://www.zigbee.org/ (visited on 30/08/2018).

ERKLÄRUNG GEMÄSS §9 DER PROMOTIONSORDNUNG

Hiermit versichere ich, die vorliegende Dissertation selbstständig und nur unter Verwednung der angegebenen Quellen und Hilfsmittel verfasst zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, November 2018

Markus Miettinen