

Seamless Authentication For Ubiquitous Devices

Shrirang Mare

Technical Report TR2016-793
Dartmouth Computer Science

SEAMLESS AUTHENTICATION FOR UBIQUITOUS DEVICES

A Thesis

Submitted to the Faculty

in partial fulfillment of the requirements for the

degree of

Doctor of Philosophy

in

Computer Science

by

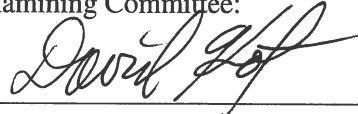
Shrirang Mare

DARTMOUTH COLLEGE

Hanover, New Hampshire

May 2016

Examining Committee:



(chair) David Kotz, Ph.D.



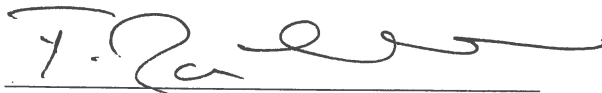
Sean Smith, Ph.D.



Lorenzo Torresani, Ph.D.



Carl Landwehr, Ph.D.



F. Jon Kull, Ph.D.

Dean of Graduate Studies

Copyright © 2016, Shrirang Mare

All rights reserved

Abstract

User authentication is an integral part of our lives; we authenticate ourselves to personal computers and a variety of other things several times a day. Authentication is burdensome. When we wish to access to a computer or a resource, it is an additional task that we need to perform – an interruption in our workflow. In this dissertation, we study people’s authentication behavior and attempt to make authentication to desktops and smartphones less burdensome for users.

First, we present the findings of a user study we conducted to understand people’s authentication behavior: things they authenticate to, how and when they authenticate, authentication errors they encounter and why, and their opinions about authentication. In our study, participants performed about 39 authentications per day on average; the majority of these authentications were to personal computers (desktop, laptop, smartphone, tablet) and with passwords, but the number of authentications to other things (e.g., car, door) was not insignificant. We saw a high failure rate for desktop and laptop authentication among our participants, affirming the need for a more usable authentication method. Overall, we found that authentication was a noticeable part of all our participants’ lives and burdensome for many participants, but they accepted it as cost of security, devising their own ways to cope

with it.

Second, we propose a new approach to authentication, called *bilateral authentication*, that leverages wrist-wearable technology to enable seamless authentication for things that people use with their hands, while wearing a smart wristband. In bilateral authentication two entities (e.g., user's wristband and the user's phone) share their knowledge (e.g., about user's interaction with the phone) to verify the user's identity. Using this approach, we developed a seamless authentication method for desktops and smartphones. Our authentication method offers quick and effortless authentication, continuous user verification while the desktop (or smartphone) is in use, and automatic deauthentication after use. We evaluated our authentication method through four in-lab user studies, evaluating the method's usability and security from the system and the user's perspective. Based on the evaluation, our authentication method shows promise for reducing users' authentication burden for desktops and smartphones.

Acknowledgments

I would like to thank everyone who helped me get to this point.

First and foremost, I am greatly indebted to my advisor David Kotz for helping me every step of the way, from encouraging me to pursue a Ph.D. to keeping me focused and pushing me to finally finish the dissertation. He imparted countless research and life skills along the way, in addition to reinforcing the value of hard work and helping me learn to write English all over again. I was very fortunate to have him as my advisor.

I was also fortunate to work with Mary Baker, Andrés Molina-Markham, Jacob Sorber, Minh Shin, and Cory Cornelius, who taught me many things and have shaped my thinking in many ways. I am grateful to Ron Peterson, who has been an invaluable source of technical expertise. I thank my committee, Sean Smith, Lorenzo Torresani, and Carl Landwehr, for their patience and feedback. I also thank Jason Reeves and Sikandar Mashayak for providing helpful feedback on the early drafts of the dissertation.

I am grateful to my close friend Aarathi Parsad for always being there to provide encouragement when I needed one and to celebrate little victories. I thank my labmates, Tim Pierson, Travis Peters, and Tianlong Yun for providing a fun and supportive environment. I thank all of my friends in Dartmouth for helping me adjust to life in a new country. In

particular, Yash Patankar, Rima Murthy, Ashok Holla, Ranganath Kondapally, Umang Bhaskar, Priya Natarajan, Paritosh Kavathekar, and Chrisil Arackaparambil. I am grateful to my host family, Terry and Betsy Cantlin, for providing me a home away from home.

My research depended on a large suite of open source software. I am indebted to the entire open source community, in particular the developers of $\text{T}_\text{E}\text{X}/\text{L}_\text{A}\text{T}_\text{E}\text{X}$, Python, IPython, Numpy, Scipy, Matplotlib, Pandas, and Scikit-learn projects.

Above all I am grateful for the unwavering support and constant encouragement from my family. My father Bhanudas taught me the value of hard work and the importance of education, my mother Anusaya always kept me grounded by reminding me the important things in life, my siblings Pandurang and Vaishali paved way for higher education and encouraged me to pursue engineering, and my wife Radha cared for me and supported me through the final year of my Ph.D. I love you all.

This research results from a research program at the Institute for Security, Technology, and Society at Dartmouth College, supported by the National Science Foundation under Grant Award Number CNS-1329686 and by the Department of Health and Human Services under award number 90TR0003-01.

Contents

Abstract	ii
Acknowledgments	iv
Contents	vi
List of Figures	xiii
List of Tables	xv
List of Abbreviations	xvii
1 Introduction	1
1.1 Vision	3
1.2 Approach	4
1.3 Challenges	5
1.4 Contributions	6
1.5 Outline	8
2 Background	10

2.1	Definitions	13
2.2	Assumptions	15
2.3	Adversary model	16
2.4	Design goals	17
2.5	Evaluation metrics	18
3	Related Work	23
3.1	History	23
3.2	Passwords	24
3.3	Hardware tokens	25
3.4	Biometrics	26
3.4.1	Physical	26
3.4.2	Behavioral	26
4	People's Authentication Behavior	28
4.1	Introduction	29
4.2	Related work	32
4.3	Study goals	34
4.4	Methodology	35
4.4.1	Authentication event	35
4.4.2	Wearable digital diary	37
4.4.3	Study procedure	43
4.5	User study participants	45
4.6	Study Limitations	45
4.7	Findings	48

4.7.1	Digital vs. physical authentication	48
4.7.2	Authentication targets and authenticators	50
4.7.3	Authentication failures	52
4.7.4	Passwords	55
4.7.5	Authentication behavior across age and gender	60
4.7.6	Guessing authentication events	61
4.7.7	Authentication patterns	61
4.7.8	Opinion about authentication	64
4.7.9	Best and worst authenticators	65
4.7.10	Things participants carry	67
4.7.11	Privacy	71
4.8	Discussion	72
4.8.1	What would we do differently?	72
4.8.2	What went well?	73
4.9	Summary	74
5	Bilateral Verification	76
5.1	A different approach to user verification	77
5.2	Security of bilateral verification approach	79
5.3	Benefits and limitations	80
5.4	Related work	81
6	Desktop Authentication	83
6.1	Overview	86
6.2	Initial authentication	88

6.2.1	Intuition	89
6.2.2	Protocol	92
6.2.3	Identifying candidate wristbands	95
6.2.4	Correlation step	96
6.2.5	Authentication feedback	100
6.3	Evaluation of Initial authentication	101
6.3.1	Feasibility user study	101
6.3.2	Quickness	103
6.3.3	Recognizing intentionality	103
6.3.4	Usability	105
6.3.5	Security	110
6.3.6	Users' perception of CSAW	112
6.4	Continuous authentication	117
6.4.1	Intuition	117
6.4.2	Architecture	119
6.4.3	Interaction extractor	120
6.4.4	Segmenter	124
6.4.5	Features	125
6.4.6	Interaction classifier	127
6.4.7	Correlator	127
6.5	Evaluation of Continuous Authentication	128
6.5.1	User study	129
6.5.2	Usability	131
6.5.3	Security	133

6.5.4	User-agnostic	135
6.6	Deauthentication	140
6.7	Discussion	142
6.7.1	Deployment issues	143
6.7.2	Other attacks	146
6.7.3	Extension to laptops	148
6.8	Summary	148
7	Smartphone Authentication	150
7.1	Overview	152
7.1.1	Challenges	154
7.2	Method	155
7.2.1	Grip detector	158
7.2.2	M2MC	160
7.2.3	M2IC	162
7.2.4	Confidence Scorer	163
7.2.5	Policy Decisions	164
7.2.6	Confidence booster	164
7.3	Evaluation	165
7.3.1	Data collection	165
7.3.2	Effectiveness	167
7.3.3	Security	176
7.3.4	Usability	178
7.4	Discussion	178
7.5	Summary	180

8	Comparative evaluation	182
8.1	Benefits	183
8.1.1	Usability benefits	183
8.1.2	Security benefits	184
8.2	Evaluation of authentication schemes	185
8.2.1	Passwords	185
8.2.2	PICO	186
8.2.3	Proximity	187
8.2.4	Fingerprint recognition	189
8.2.5	Voice recognition	189
8.2.6	Face recognition	190
8.2.7	Impedance	191
8.2.8	Keystroke dynamics	192
8.2.9	Touch input dynamics	193
8.3	Conclusion	193
9	Summary and Future work	196
A	Authentication behavior user study	202
A.1	Pre-logging interview	202
A.2	Post-Logging interview	203
B	Desktop user studies	205
B.1	Interview questions	205
B.2	SUS Survey	206

List of Figures

2.1	Authentication decision	19
2.2	Confusion matrix	20
4.1	Subset of an individual's authenticators	30
4.2	Watch on the wrist	38
4.3	Watch app UI	40
4.4	Phone app UI	42
4.5	Study participant demographics	46
4.6	Authentication events logged by participants	49
4.7	Failure rate for different authentication targets	55
4.8	Failure rate for different authenticators	56
4.9	Password failure rates for different authentication targets	57
4.10	Average authentication events per day across participants' age	60
4.11	Distribution of authentication events across weekday and weekend	62
4.12	Distribution of authentication events across hours in a day	63
6.1	Wrist data sample	87

6.2	Desktop input sample	87
6.3	CSAW block diagram for desktops	88
6.4	Authentication protocol	93
6.5	Wrist acceleration for Tap-5x interaction	98
6.6	FNR for participants	107
6.7	FNR and FPR for simultaneous authentication attempts	109
6.8	FNR and FPR for self mimicking	112
6.9	SUS score comparison	116
6.10	Wrist acceleration during computer use	119
6.11	CSAW architecture for continuous authentication	120
6.12	Keyboard zones	121
6.13	Average FNR vs. window size	132
6.14	Average FPR when user is walking nearby	134
6.15	Average FPR when user is writing nearby	134
6.16	Average FPR when adversary is using a desktop nearby	136
6.17	Probability of a user retaining access at time t	138
6.18	Probability of identifying an adversary at time t	139
6.19	Probability of identifying an adversary at the end of window w	140
7.1	CSAW architecture	157
7.2	Wristband orientations during phone use.	159
7.3	Instantaneous confidence score for a participant	171
7.4	Wristband acceleration relative to the phone for different grips	172
7.5	Effect of sampling rate on BAC	173
7.6	Effect of window size on BAC	175

List of Tables

2.1	Common statistical measures for a binary decision system:	21
4.1	Distribution of authentication events by targets	50
4.2	Distribution of authentication events by authenticators	52
4.3	Failure rates for different targets and authenticators	53
4.4	Distribution of authentication events by location	63
4.5	Participants opinion regarding authentication	64
4.6	Authenticators liked by the participants	66
4.7	Authenticators disliked by the participants	67
4.8	Authenticators carried by the participants	68
6.1	FPR for detecting user's intent to authenticate	105
6.2	FNR and FPR for different window sizes	136
7.1	FPR and BAC for initial authentication with pick-up action	169
7.2	FPR and BAC for initial authentication with rotate action	169
7.3	FPR and BAC for continuous authentication	171

7.4	FPR, FNR, BAC for different subsets of features and sensors	176
8.1	Comparative evaluation of CSAW and other authentication schemes	195

List of Abbreviations

List of abbreviations used in this dissertation, with page numbers where the abbreviation is first used.

BAC	Balanced Accuracy	21
BLE	Bluetooth Low Energy	179
CSAW	Continuous Seamless Authentication using Wristbands	5
CTSS	Compatible Time-Sharing System	24
DDTS	Dartmouth Time-Sharing System	24
EWMA	Exponentially Weighted Moving Average	163
FNR	False Negative Rate	21
FN	False Negative	19
FPR	False Positive Rate	21
FP	False Positive	19
IRB	Institutional Review Board	129
M2IC	Motion-to-Input Correlator	156

M2MC	Motion-to-Motion Correlator	156
RFID	Radio-frequency Identification	78
ROC	Receiver Operating Characteristics	21
RSS	Received Signal Strength	82
SUS	System Usability Scale	113
TNR	True Negative Rate	21
TN	True Negative	19
TPR	True Positive Rate	21
TP	True Positive	19
UDS	Usability Deployability Security	182
ZIA	Zero-Interaction Authentication	187

1

Introduction

We perform authentication several times everyday: unlocking a smartphone (with a PIN or a fingerprint), logging in to a laptop or a desktop (with a password), unlocking a car (with a key), and logging in to a website (with a password); in the near future, we may also authenticate to smart devices such as smart TV remote control, power tools, medical devices, exercise equipment, and electronic doors. All these are instance of *user authentication*. Formally, user authentication is the process of verifying the identity of the user, so that we

can grant access to a resource only to the rightful (authorized) user. In this dissertation, we focus on user authentication for desktops, laptops, smartphones, and tablet computers, henceforth commonly referred as *computers*, unless otherwise specified.

Today, computers are almost ubiquitous: according to a 2015 survey in the United States, conducted by the Pew Research Center, about 70% of adults own a desktop or a laptop; about 68% of adults own a smartphone; and about 45% of adults own a tablet [69]. Indeed, it is clear that smartphone and tablet ownership will continue to increase in coming years. People are increasingly storing sensitive information on their computers, especially on their smartphones that provide easy access to the information, but they are not using adequate measures to protect their computers. In the United States, 40% of smartphone users do not use a PIN or a passcode to secure their smartphone [49], and many others use simple 4-digit PINs, which are easy to steal.

To understand why people do not use adequate security measures, we need to look closer at how people use their computers: people access their computers numerous times in a day, often just for a few minutes at a time. In 2010 in a study of 17,000 BlackBerry users, Oliver found that BlackBerry users interacted with their smartphones about 87 times per day on average, and the average duration of each interaction was 68 s [65]; in a more recent (2013) study of 16,000 Android users, Wagner et al. found the average interactions per day to be about 57, and the average interaction duration to be about 115 s [98]. Typically, people access their laptops and desktops less frequently than their smartphones, but depending on the occupation, some people may access their work computers much more than others; in a busy hospital, clinicians and nurses access their work computer terminals numerous times a day – “about 100 to 200 times a day”, quoted a clinician we asked. Given how frequently individuals access their computers, if they find an authentication method burdensome, they

are less likely to use it.

The most common authentication method – passwords – is indeed burdensome for many people: to use a password, the user has to first recall it (mental effort) or find it (physical effort), and then type the password (physical effort). The authentication process requires time and effort, and affects users’ working memory, disrupting their workflow. Adams and Sasse showed that people devise workarounds when they are forced to use an authentication method that does not mesh with their workflow [1]; Sinclair and Smith observed similar behavior in hospitals and corporate enterprises [85, 86].

Common workarounds include writing down passwords, sharing passwords (if the user forgets, others may remember), or leaving the computer unlocked to avoid having to authenticate next time. Leaving a computer unattended and unlocked raises an interesting problem of *deauthentication*: how to de-authenticate a user from a computer when she is no longer using that computer. This is an important aspect of an authentication scheme – “Deauthentication is one of our biggest vulnerabilities”, said the Director of Information Systems at a hospital we visited – but unfortunately ignored by most authentication methods. These workarounds may reduce the authentication burden on users, but they leave the computer vulnerable. To deter such workarounds, we need an effortless authentication method that blends seamlessly in to people’s workflow, so that they would be more likely to use it.

1.1 Vision

The authentication method we envision supports all three aspects of authentication: *initial authentication*, to verify the individual’s identity before she can access a computer; *continuous authentication*, to continuously verify the authenticated individual while she uses the

computer; and *deauthentication*, to de-authenticate the individual when she stops using the computer. The method provides a quick and effortless initial authentication so an individual can start using the computer for the intended task without any disruption to his/her workflow, automatic deauthentication so that the computer is secured when the authenticated individual is not using/attending the computer, and passive continuous authentication so that it does not disrupt the individual's task.

Our envisioned method blends into people's workflow seamlessly and appears invisible to them. In our envisioned method, to use a desktop computer, Alice simply walks up to the computer, turns on the display, and starts using the computer, for her intended task. Alice does not perform any task explicitly for authentication; the computer authenticates Alice using the first few seconds of interaction (input from Alice), continues to authenticate her while she uses the computer, and deauthenticates her when she leaves. Similarly, to use a smartphone, Alice simply picks up her smartphone (from a table or her purse) and starts using it; the smartphone authenticates Alice when she picks it up, continues to authenticate her while she uses it, and deauthenticates her when she puts the phone back (on the table or in her purse).

1.2 Approach

To develop an authentication method that blends with people's workflow seamlessly, we start with their workflow: everyone interacts with a computer when performing a task on the computer, and the interaction involves providing input to the computer, usually with their hands. In our approach, we use these inputs to authenticate an individual, but unlike behavioral biometrics (such as keystroke dynamics) our approach does not involve learning any characteristics unique to an individual. Our approach involves the individual

(say, Alice) wearing a wristband that can sense her wrist motion and communicate with the computer. When Alice provides input to the computer using her wristband hand, the computer authenticates Alice if her wrist movement (when she provided the computer input) correlates with the provided input. Computer interaction when performed with the wristband hand is a result of the individual's wrist movement – for example, when Alice provides input with a keyboard, her wrist exhibits ‘typing’ movement; when she provides input with a mouse, her wrist exhibits a ‘mousing’ movement pattern; when she picks up her phone or holds her phone in her wristband hand, the wristband exhibits movement similar to the phone. Thus, the computer authenticates the individual if her wrist shows the expected movement that correlates with the computer input. We call this approach *bilateral verification*, because it involves two parties – the computer and the wristband – working together to verify that the input indeed came from the individual. We describe the general approach in Chapter 5; based on our approach we develop an authentication method called Continuous Seamless Authentication using Wristbands (CSAW) for desktop and smartphone authentication, which we describe in Chapter 6 and Chapter 7, respectively.

1.3 Challenges

There are several challenges in realizing the above vision with our approach: How to blend the authentication process into people's workflow so that it becomes effortless? Which inputs (user-computer interaction) to use for authentication? What behavior to expect for different inputs and how does the computer correlate wrist movement with these inputs? How to keep the authentication process invisible yet conforming to people's mental model? How to implicitly detect an individual's intent to authenticate? How does the computer recognize its user, among the multiple authorized individuals (with wristbands) in radio proximity?

How is the wristband tied to its owner’s identity, i.e., how does the wristband authenticate its wearer? How and when to deauthenticate the user? How do computers communicate with multiple wristbands at the same time? How to implement the authentication method in the wristband in an energy efficient manner?

Some challenges – such as the securely linking a wristband to its wearer – can be achieved with existing solution: Cornelius et al. use an impedance biometric to recognize the wearer of a wristband [17], and Apple Watch requires (if the associated security feature is enabled) its wearer to enter a PIN to activate the watch [7]. Some challenges (such as an energy-efficient implementation of the approach) are out of scope of this dissertation, because such an implementation would require hardware support.

In this dissertation we address three main challenges that take us a step closer towards our vision of seamless authentication:

1. How to authenticate an individual with minimal effort and without disrupting her workflow?
2. How to passively and continuously verify an authenticated individual while she uses the computer?
3. How to automatically deauthenticate the individual when she stops using the computer?

1.4 Contributions

We make four contributions in this dissertation.

First, we conducted a user study with twenty-six participants to understand their authentication behavior. The study was conducted in three parts: first, we interviewed the participant about his/her behavior; second, we asked the participant to log his/her authentications – to

physical things (e.g., doors, cars) and digital things (e.g., computers, websites) – for one week; and third, we conducted a final exit interview with follow-up questions from the first interview and from the logged data. Chapter 4 discusses this study and its results. Part of this work is published in the Symposium on Usable Privacy and Security (SOUPS) 2016 [51].

Second, we propose a new approach for authentication, called bilateral verification. Most current authentication methods authenticate an individual by comparing a stored secret value associated with the individual’s identity with a value the individual presents at the time of authentication; for example, in password-based authentication, the individual is authenticated by comparing the password s/he enters at the time of authentication with the stored password s/he once chose. In bilateral verification, the individual is authenticated by comparing two values, but both values are measured during authentication and discarded after authentication. In Chapter 5 we formalize the approach and discuss its advantages and disadvantages.

Third, based on our approach, we developed an authentication method for desktop users that does initial authentication, continuous authentication, and deauthentication. In this method an individual authenticates to a desktop computer by simply tapping a key five times; the desktop continues to verify the authenticated individual as she provides more keyboard and mouse inputs; and when the individual walks away from the desktop, she is deauthenticated automatically. Chapter 6 discusses this method and our evaluation. Part of this work is published in the IEEE Symposium on Security and Privacy (S&P) 2014 [52].

Fourth, we developed an authentication method for smartphone users, also based on the bilateral verification approach, and along the same philosophy – to make the authentication step invisible to the user. In this method we leverage the fact that smartphones and wristbands have motion sensors: when an individual picks up the phone or holds it in the wristband hand, the phone and the wristband would have similar motion. When the individual is not

holding the phone in her wristband hand, we use her touchscreen inputs to correlate with her wrist movement and to authenticate her. Chapter 7 discusses these methods and our evaluation.

1.5 Outline

The remainder of this dissertation is organized as follows.

Chapter 2 provides the required background for this dissertation: definition of the terms we use, the desired security and usability goals, assumptions we make for our method, the adversary model, and the evaluation metrics we use to measure the effectiveness of our authentication method.

Chapter 3 presents the related work, providing an overview of the history of authentication schemes for desktops and smartphones, ranging from passwords to some of the recently proposed authentication schemes.

Chapter 4 describes the methodology and findings of a user study we conducted to study users' authentication behavior. The user study involved semi-structured interviews with twenty-six participants about their everyday authentication behavior, and each participant self-logged, in the moment, all the authentication events s/he performed during the one week of the study.

Chapter 5 introduces bilateral verification, a new approach to verify the user of a computer. We present a generalized bilateral verification approach, and discuss its advantages and disadvantages compared to the traditional verification approach.

Chapter 6 and Chapter 7 describe our authentication method for desktops and smartphones, respectively, based on the bilateral verification approach. In these chapters we describe the authentication method, authentication protocol, and the methodology and results

from our experiments and user studies.

Chapter 8 provides a comparative evaluation of CSAW with the related authentication schemes; the focus of evaluation is on usability and security of the schemes.

Finally, Chapter 9 summarizes our work and discusses immediate extensions and more future work.

2

Background

Authentication is the act of verifying the truth of an attribute claimed to be true by an entity. The word *authenticate* originates from medieval Latin *authenticāt-*, which means ‘established as valid’. In computing, authentication is used in the context of users, processes, or data; in this dissertation, we use authentication in the context of users – the act of verifying the identity of a user.

Authentication has two components: user identification and user verification, i.e., verifi-

cation of the user's identity. When we present a photo ID to an officer for authentication, the name on the ID card provides user identity and the photo is used for verification; when we log in with a username and password, the username provides user identity and the password is used for verification. Some authentication methods only perform user verification and require the individual to provide her identity (e.g., in a password-based method, the user provides her identity when she enters her username); some authentication methods may perform both user identification and user verification (e.g., a fingerprint can be used to identify and verify an individual). Our method performs both user identification and user verification. In our approach, the wristband provides the identity of its wearer, and in this dissertation, we address the verification problem – how to verify whether an individual (wristband wearer) is using a given computer.

Authentication involves two entities, an individual and a computer; the individual authenticates to the computer. Depending on whether the computer is shared by individuals and whether an individual has access to multiple computers, there are three scenarios:

- *one-to-one*, which involves a single user and single computer, e.g., an individual using his/her personal phone;
- *many-to-one*, which involves multiple users sharing a single computer, e.g., a desktop shared at home by family members; and
- *many-to-many*, which involves multiple users sharing multiple computers, e.g., clinical staff sharing computer terminals in a hospital.

The one-to-one scenario is the simplest of the three – there is no need to identify the user, because there is only one user; the other two scenarios add some complexity to the authentication problem. In many-to-one, the authentication method needs to identify

and verify the right user; in many-to-many, the authentication method should prevent an individual from being authenticated to multiple computers at the same time, unless it is desired. The following use cases exemplify these scenarios.

Nancy. At work, Nancy makes it a point to lock her desktop every time she leaves her cubicle, even if it is only for a few minutes. She has memorized a strong password, compliant with the company IT policy, and she is used to entering it multiple times throughout a day. But she wishes there was an easier authentication method that did not require her to make the effort of recalling and typing her long password, and would quickly lock her desktop when she steps away, so that she can have peace of mind when she leaves the cubicle and can resume her work when she is back at her desktop without the mental context switch to recall her password.

Jane. At home, Jane shares a laptop with her family, including two teenagers. Jane also uses a strong password, because she worries that her clever teenagers may learn her password by shoulder surfing, and she remembers to log out, because she does not want anyone to accidentally use her account, which she also uses for her work.

Smith Memorial Hospital. At Smith memorial hospital, clinicians use desktop terminals to access patient records. The terminals are placed in patient units and other areas frequented by clinical staff. All the terminals are shared by all clinicians; clinicians have to log in to a terminal with their username and password, and are expected by the IT people at the hospital to log out when they leave. The IT staff worries that an unattended logged-in terminal can lead to errors in patient record entry or a breach of patient privacy, if the terminal is accessed by an unauthorized passerby. Due to the nature of their workflow, clinicians need to access

many patient records at different times of the day and from different terminals in the hospital. Nancy and her clinical colleagues wish there was an easier authentication method they could use that did not require them to specify their username and password every time they want to use a terminal, and that automatically logs them out when they leave, so they can focus on patient care. (Indeed, we repeatedly hear these very desires from many clinical staff we meet.)

These are the types of use cases we envision where our approach would be most effective. In Chapter 3 we revisit these use cases and discuss the advantages our approach offers over other methods.

2.1 Definitions

As mentioned above, the authentication problem involves two entities, an *individual* who attempts to authenticate to a *computer*; if successful, the individual uses the computer to perform certain tasks. In this context, we define the terms we use in this dissertation.

Computer: a desktop computer or (in some of our work) a smartphone.

Target computer: the computer of interest in the authentication problem, i.e., the computer that an individual wishes to access. In the context of an adversary, the computer to which the adversary wants access.

User: an individual who is authorized to use the target computer.

Adversary: an individual who attempts to access the target computer in an unauthorized manner. The adversary can be a user of the target computer, in which case he acts as an adversary if he intentionally attempts to access the computer by impersonating a

different user. To improve readability, without loss of generality, we consider a *user* to be a female and an *adversary* to be a male.

Victim: a user who the adversary targets; in other words, a user whose account the adversary wants to access on the target computer.

Interaction: the inputs provided by the user to the computer via input interfaces (e.g., keyboard, mouse, touchscreen) and sensors in the computer (e.g., motion sensors in a smartphone or tablet), or the output from the computer consumed by the user. An interaction can be *passive*, where the user only consumes output from the computer, or *active*, where the user only provides input to the computer. In this dissertation we consider only active interactions and the term *interaction* implies *active interaction*, unless explicitly specified otherwise.

Session: a session is the time duration when the user is using the target computer.

Initial authentication: the process of authenticating an individual to a computer before the start of a session – for example, unlocking a smartphone or logging in to a desktop or laptop. It is important to distinguish between initial authentication and continuous authentication. Initial authentication refers to verifying the user’s identity *once* to allow her access to the computer; a user cannot use the computer without a successful initial authentication.

Continuous authentication: the process of authenticating the current user during a session, after a successful initial authentication. Continuous authentication serves two purposes: *a*) as a backup if the initial authentication fails to prevent an adversary from authenticating, and *b*) to provide high assurance about the user’s identity, if required, say, for a sensitive application or service on the computer.

Deauthentication: the process of de-authenticating a previously authenticated user at the end of her session. Deauthentication is the reverse of authentication; it refers to removing the user’s access to the computer so that if the user wants to use it again, she has to start with initial authentication.

2.2 Assumptions

Every security solution is based on some assumptions. We make the following assumptions regarding the user of CSAW and the hardware and the software used in CSAW.

- A1 *Wristband hardware:* The wristband has a built-in accelerometer and gyroscope sensors, and a radio that can communicate securely with the target computer. These technologies are common in fitness bands and smartwatches today, such as Android Wear [4], Apple Watch [7], FitBit [27], Jawbone UP [38], Misfit [57], and Pebble [67]; we envision CSAW being integrated into similar devices. We assume the wristband has resources to use encryption for its communication with the target computer. We also assume that the wristband can contain identity information sufficient to identify its wearer and that this information cannot be stolen or compromised by an adversary.
- A2 *Wristband is worn:* The user wears her wristband on one of her wrists, called the ‘wristband hand’; either hand is fine. To use CSAW, the user should interact with the target computer using her wristband hand.
- A3 *Wristband is paired:* The wristband is already paired with the target computer, a one-time activity, using a secure pairing method [42]. All communication between the wristband and the computer is encrypted – using the keys shared during pairing – for confidentiality, integrity, and mutual authenticity. In an enterprise setting, we assume administrative tools pair all wristbands with all computers in a distributed fashion.

- A4 *Wristband is personal*: Users do not share their wristband with others, because in CSAW the wristband serves as its owner's (user's) identity token. If desired, one could securely link the watch to its owner (to prevent any unintended sharing) by detecting when the watch is removed and requiring the wearer, when she dons the wristband, to enter a password – a security feature supported by Apple Watch [7]; or using a biometric to identify the owner, as shown by Cornelius et al. [17]. Note that this “personal token” assumption is similar to the use of ID cards that many organizations have.
- A5 *Synchronized clocks*: The wristband and the target computer clocks are synchronized to 10 msec resolution or better. Some of our methods use temporal correlation, so it is important that the wristband and the computer clocks are synchronized. The computer and the wristband synchronize their clocks using a suitable clock synchronization algorithm for one-hop low-power wireless networks [89].

2.3 Adversary model

We consider an adversary who is a curious family member or friend, a curious colleague, or a malicious individual with physical access to the target computer. The adversary's goal is to access the victim's account on the target computer; the adversary can be an authorized user of the target computer, in which case, he intends to access the target computer masquerading as another user. We make the following assumptions about our adversary's capabilities.

- C1 *Observe the victim*: The adversary can observe (and video-record) the victim authenticating to her computer or when she is using her computer. The adversary can also observe the victim and her wrist when she is in radio proximity of the target computer.
- C2 *Physical access*: The adversary has physical access to the target computer when the

victim steps away and leaves her computer behind. The adversary may be another authorized user of the computer.

- C3 *Computationally bounded*: The adversary does not have the encryption keys exchanged by the wristband and the computer during their pairing, and he is computationally bounded so he cannot break the encrypted communication between the wristband and the computer.
- C4 *Hardware and software*: The adversary cannot compromise the components that CSAW relies on to authenticate users – software and hardware on the wristband, the computer’s input interfaces, and the CSAW software on the computer.
- C5 *Victim’s wristband*: The adversary cannot use the victim’s wristband and authenticate as the victim user. Thus, the adversary cannot pose as the victim user by spoofing the victim’s wristband or by stealing and wearing the victim’s wristband. (These threats have well-known solutions, complementary to CSAW, as we mentioned in A3.)

We are not concerned about an attacker who performs jamming attacks that would prevent legitimate communication between the wristband and the computer. We are primarily concerned with an impersonation attack, where the adversary successfully fools CSAW and gets access to the target computer masquerading as the victim user. The strategy that the adversary chooses for his impersonation attack depends on the specific authentication method implementation for the target computer. We discuss these strategies when we discuss CSAW for desktops and smartphones in Chapter 6 and Chapter 7, respectively.

2.4 Design goals

We desire the following security and usability goals for CSAW. We use these goals to evaluate CSAW and compare it with other related authentication methods.

- G1 *Effortless*: The authentication process should not require physical effort beyond, say, pressing a button. Users should not have to remember any secret for authentication.
- G2 *Quick*: The time the user must spend for each authentication should be short, say, within 1-2 seconds. The setup time for the authentication method, a one-time task, can be longer than authentication time.
- G3 *Intentional*: The authentication process should start only when the user intends to use the computer. The intent to use can be implicit, such as turning on the display of the computer or pressing a button.
- G4 *Usable*: Users should not encounter errors frequently; the authentication method should have a low error rate for authenticating users.
- G5 *Secure*: An adversary should find it hard to gain access with the method; the authentication method should have a low error rate for allowing unauthorized access.
- G6 *Continuous*: The authentication method should continuously verify the user when she is using the computer, without any effort from the user.
- G7 *User-agnostic*: The authentication method should not depend on the user's individual characteristic, physiological (e.g., fingerprint) or behavioral (e.g., typing behavior).

2.5 Evaluation metrics

An authentication method is essentially a decision algorithm that takes credentials from an individual (when the individual attempts to authenticate) and outputs a decision whether the individual should be authenticated. Figure 2.1 shows an authentication method that outputs a binary decision from the set $\{P, N\}$: P (positive) if the authentication is successful, otherwise N (negative); alternatively, an authentication method could output a continuous value (between 0 and 1) indicating the probability that the individual should be authenticated,

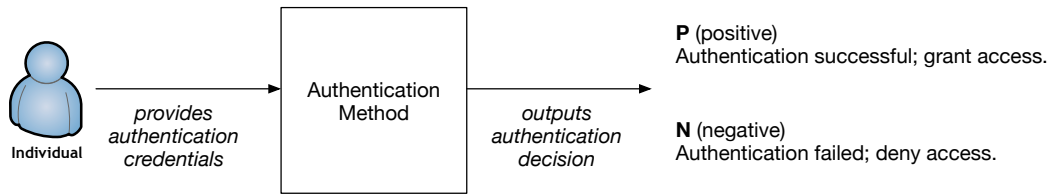


Figure 2.1: An authentication method takes authentication credentials from an individual and outputs either a **P** (positive, on successful authentication) or a **N** (negative, on failed authentication).

and let the application decide whether to authenticate the individual (and allow access to the resource), but, for simplicity, we describe evaluation metrics to measure the effectiveness of authentication methods that output binary decisions.

For a given authentication attempt (called a *test case*), the decision that the authentication method *should* output is called the *true value*, and the decision that the authentication method *actually* outputs is called the *predicted value* – the method’s prediction of the true value. Thus, given a test case, one of four things can happen. If the individual should be authenticated (a positive test case) and the method outputs positive, it is counted as a True Positive (TP); if the method outputs negative, it is counted as a False Negative (FN). If the individual should *not* be authenticated (a negative test case) and the method outputs negative, it is counted as a True Negative (TN); if the method outputs positive, it is counted as a False Positive (FP). Given a set of test cases, a two-by-two *confusion matrix* can be formed representing the outcome of the tests. Figure 2.2a shows the confusion matrix with these four outcomes, and Figure 2.2b shows an example confusion matrix for 100 test cases – 10 attempts by a user and 90 attempts by an adversary. In the example, out of the 10 attempts, the user succeeds 9 times and fails once; out of the 90 attempts, the adversary fails (as he should) 88 times, but succeeds twice.

The numbers along the major diagonal of the confusion matrix (TP and TN) represent the correct decisions made by the authentication method, and the numbers along the minor

		True value	
		P	N
Predicted value	P	True Positive	False Positive
	N	False Negative	True Negative

(a)

		True value	
		P	N
Predicted value	P	9 TP	2 FP
	N	1 FN	88 TN

(b)

Figure 2.2: Confusion matrix. (a) standard confusion matrix and (b) a confusion matrix with 10 authentication attempts from a user, among which 1 attempt failed (FN) and 9 succeeded (TP); and 90 authentication attempts from an adversary, among which 88 failed (TN), as they should, but 2 attempts were successful (FP).

diagonal (FN and FP) represent the errors – the confusion – of the method. This confusion matrix forms the basis for many common metrics. Table 2.1 shows some of these common metrics; for other commonly used statistic measures, see the discussion of different evaluation metrics by Fawcett [26] and Powers [72].

Depending on the field (e.g., biometrics, information retrieval, medicine), different measures are used to evaluate the effectiveness of the underlying method, and the measures also have field-specific names; hence, some measures have multiple names. *Specificity* and *sensitivity* are common in behavioral sciences and medicine; *precision* and *recall* are common in information retrieval. In computer security, *false positive rate* and *false negative rate* are commonly used to evaluate an authentication method; these measures are called *false accept rate* and *false reject rate* in biometrics. We use the following metrics to evaluate the effectiveness of CSAW.

False Negative Rate (FNR) tells us how effective a method is at authenticating users, in other words, how frequently the method makes an error and denies access to a user; we want this number close to zero. A poor method that makes frequent errors will

Table 2.1: List of common statistical measures for a binary decision system, their formulas, and their other names.

Measure	Formula	Also Known As
False Negative Rate (FNR)	$\frac{FN}{TP + FN}$	false reject rate, miss rate
False Positive Rate (FPR)	$\frac{FP}{TN + FP}$	false accept rate, fall-out
True Positive Rate (TPR)	$\frac{TP}{TP + FN} = 1 - \text{FNR}$	recall, sensitivity, true accept rate
True Negative Rate (TNR)	$\frac{TN}{TN + FP} = 1 - \text{FPR}$	specificity, true reject rate
Positive Predictive Value	$\frac{TP}{TP + FP}$	precision
Negative Predictive Value	$\frac{TN}{TN + FN}$	
Accuracy	$\frac{TP + TN}{TP + TN + FP + FN}$	
Balanced Accuracy (BAC)	$\frac{0.5 \times TP}{TP + FN} + \frac{0.5 \times TN}{TN + FP}$	

cause user frustration; thus, this is one measure of the method's usability.

False Positive Rate (FPR) tells us how effective a method is at stopping adversaries, in other words, how frequently the method makes an error and accidentally grants an unauthorized individual access to the computer; we want this number close to zero. A poor method that makes frequent errors is a security risk; thus, this is one measure of the method's security.

True Positive Rate (TPR) $= 1 - \text{FNR}$, is another measure of how effective a method is at authenticating users; we want this number close to one. We use TPR and FPR to plot the Receiver Operating Characteristics (ROC) curve and analyze different parameters in our authentication method.

Balanced Accuracy (BAC) tells us the overall accuracy of the method by taking into account any imbalance (number of positive vs. negative test cases) in the test data; we want this number to be close to one. Often in testing, there is a high imbalance between

the test cases: either the number of positive test cases is significantly greater than the negative test cases, or vice-versa; the accuracy metric (Table 2.1) fails to account for this imbalance and gives a false measure of the method’s effectiveness. To illustrate, in our example above (Figure 2.2b), where the test cases are imbalanced (10 positive and 90 negative), the accuracy of the method is 97%, but, if we look closely, the method’s accuracy in authenticating users is only about 90% with the method being 97.5% accurate in detecting adversaries; the BAC for this example is about 93.8%, a better measure of the method’s effectiveness than Accuracy.

We obtain test cases for our evaluation through user studies. In user studies, we ask participants to perform authentication attempts and we use those attempts as positive test cases. We also ask them to play the role of an adversary and attempt authentication using our method; we use those attempts as negative test case; we also generate negative test cases for different attack scenarios by using data from multiple participants. We elaborate on how we generate test cases when we discuss our evaluation results.

3

Related Work

This chapter provides a brief history of user authentication covering different authentication schemes.

3.1 History

The concept of using a secret word to authenticate a human dates at least as far as back to the ancient Rome where “watchwords” were used in the military to prevent infiltration

as documented by historian Polybius [71]. Human authentication to a computers began in early 1960s with the advent of multi-user computer operating systems. Since then as our needs for authentication have grown numerous authentication schemes have been proposed. Authentication schemes can be divided into three general categories, based on how they authenticate the user: what you *know* (knowledge-based), what you *carry* (token-based method), and what you *are* (biometrics); below, we discuss these categories.

3.2 Passwords

The time-sharing computer systems introduced in 1960s – the Compatible Time-Sharing System (CTSS) at MIT in 1962 and the Dartmouth Time-Sharing System (DDTS) at Dartmouth in 1964 – are often considered as the first computer systems to deploy passwords [79]. The use for passwords in these systems was derived from the need to control access to the computing resources rather than protecting secret data. These early systems did not employ strong password security mechanisms; for instance, passwords were stored in unencrypted form. People quickly discovered attacks on the password security of these systems, and system designers added security measures to counter the attacks, which resulted in techniques such as slow-encryption (to thwart brute force attempts), use of less predictable passwords, and salted passwords [59]. For resource constrained devices such as ATM machines and mobile phones, a simpler version of the passwords – PINs – were introduced. Thus, as computer systems and attacks have evolved so has our use of passwords.

People find it hard to manage passwords and there have been numerous studies documenting people’s frustrations and difficulties with password management. Passwords are, however, hard to replace as Bonneau et al. found in their evaluation of various authentication schemes [13]. The main reason being the simplicity, scalability, and deployment cost of

passwords – the cost to the system – is small. The cost to the user is, however, not small; Herley found that users do not follow the ‘good password practices’ because of these practices offer a poor cost-benefit trade-off for them [33]. Security experts are reconsidering their advice on the use of password from usability point of view [20], and there has been an increased interest from researchers and industry to seek viable (more usable) alternatives for passwords.

3.3 Hardware tokens

The idea to use a hardware token for user authentication goes at least as far back as 1987, when Spender proposed the use of handheld ‘personal authenticators’ or tokens for authentication to computers [88]. Since then researchers have proposed several token-based authentication schemes. In 1997, Landwehr proposed an authentication scheme for securing unattended computers without any software modification on the computer [43]. In 2002, Corner and Noble proposed an authentication scheme to secure laptops based on the proximity of the authorized user [18]. In 2011, Stajano proposed an authentication scheme called PICO with the goal to replace passwords [90]. Commercially, token-based authentication schemes have not been very successful, perhaps due to the involved deployment cost. Token-based authentication schemes have been, however, successful as the second factor in two-factor authentication schemes, e.g., RSA SecurID [78], YubiKey [101]. Increasingly companies are integrating their tokens in smartphones, eliminating the need to carry an additional device; this may further encourage people to use two-factor authentication.

3.4 Biometrics

Biometrics [37] are the “what you are” type of authentication schemes. They leverage the physical or behavioral characteristics across individuals.

3.4.1 Physical

Fingerprint, voice, and face are common physical characteristics used for biometric authentication. Other physical characteristics include iris, palm-print, hand geometry, body’s impedance response [17], and ear cavity shape [10]. Commercially, the use of these biometric approaches has been limited in organizations to secure valued resources, but recently biometrics have found their way in smartphones and personal computers. Apple introduced TouchID, its fingerprint authentication scheme, on iPhones in 2015 [6]; since then fingerprint authentication also being offered in several Android phones. Some consumer devices such as Android smartphones and Microsoft Surface offer facial recognition as an alternative authentication scheme (or primary if the user chooses to do so). Among these biometrics, only fingerprint seems to be used at a wide scale to secure personal computers.

3.4.2 Behavioral

Behavioral biometrics rely on the user’s behavior, i.e., *how* the user does certain things. Examples of behavioral biometrics include keystroke dynamics (how the user types), gait (how the user walks), mouse dynamics (how the user operates mouse), and touch dynamics (how the user provides touchscreen inputs). Among these characteristics, keystroke dynamics has been studied the most. The idea has its origin in the observations that telegraph operators have distinctive patterns, called *fists*, of keying messages over telegraph lines [15, 95]. Researchers have published papers on keystroke dynamics since 1980, each with their own

unique set of individuals and exploring variations in input, granularity of measurements, required training time, and so forth [58, 66]. On touchscreen devices such as smartphones and tablets researchers have proposed behavioral authentication schemes using touch input dynamics [45, 103]. Commercial use of these schemes have been limited so far. The most notable project for use of behavioral biometrics on personal computers is Google's project Abacus [29] (still in the research phase), which uses touch input dynamics to verify smartphones users.

4

People's Authentication Behavior

Before we present our authentication scheme for desktops (Chapter 6) and smartphones (Chapter 7), we discuss our findings about people's authentication behavior. During my summer internship at HP Labs in Palo Alto, California, I had the opportunity to conduct a user study to learn about people's authentication behavior. This chapter presents the methodology and findings of that study.

4.1 Introduction

Many of us carry several things such as car key, house key, corporate badge, bike key, RSA token key, bus pass, credit card, driver's license, ATM card, and so forth, with us every day to access doors, computers, and services we need; Figure 4.1 shows a subset of authentication related things carried by a person. We also use passwords, pin codes, and fingerprints to access devices, websites, and applications. These are all ways of authenticating ourselves – providing evidence that we are the right people to unlock the restricted resources around us. We expect people, especially those working in corporate environments, to carry these authentication tokens and remember complex passwords. This burden leads to *frustration* (when we forget our badges, keys, and passwords), *security breaches* (when we tailgate other people through secure doorways or write down our passwords or leave our devices unlocked), and *IT expense* (when we call help desks to reset passwords or issue new tokens). Password resets make up 10% to 30% of IT helpdesk calls, and can cost from \$50 to \$150 each to resolve [100]. Even physical keys present an increasing risk, as new smartphone apps enable scanning an unattended key in a few seconds and then printing copies of it by mail order or at kiosks [30].

Evidence suggests that password authentication can indeed be burdensome for users [20], and experts provide several approaches for addressing this problem, such as using password managers [36], but how about other forms of authentication? Is it only worth addressing the use of passwords, or are physical authentication tokens also problematic? (Yes.) How much authentication of different kinds do people actually do during the day, and does it correspond with their own concept of the burden they face? (A diverse amount, and there's no correspondence.) How failure-prone are the different kinds of authentication? (Surprisingly high.) Do people generally agree about what kinds of authentication they like and dislike, or



Figure 4.1: A subset of authentication related things carried daily by one person, who also has to manage over 250 passwords. (Photo by Mary Baker.)

will it be hard to help the bulk of people in the same way? (They do not agree, and it will be hard to make everyone happy.) We wanted answers to these and other similar questions about people’s authentication behavior, and we hope this information can be useful to others as well.

To gather a better understanding of people’s authentication behavior and ways to alleviate their possible authentication burden, we conducted a wearable digital diary user study of twenty-six people, including teenagers and adults, students, corporate employees, and others. We provided participants with a commercially available wrist wearable, the MOTOACTV [60] running our own logging application, and asked participants to log all their authentication events for a week. We used the wrist wearable to make logging quick and easy so that participants could log authentication events immediately after they performed an authentication. We designed a “slot machine” application interface to provide the wearable with an immediately available and streamlined logging process. Our hope, although we

cannot substantiate this, is that easy, immediate logging reduces the underreporting of events that can affect diary studies [46]. In addition, we are interested in authentication with physical infrastructure and not just online authentication behavior, so we could not instrument all of the participants' targeted resources to log authentication events automatically. Our results include 4,623 hours of logged events, interviews of each participant before and after the week of logging, and comments they entered through daily surveys on their smartphones. Our results thus include quantitative information in the form of "traces" of user authentication behavior as well as qualitative information in the form of participants' opinions.

Our contributions include the design and lessons learned for our wearable digital diary study, and the results of the study. Twenty-six participants is not a large enough population to allow us to make broad claims about any particular demographic, or even the general population; instead we get an in-depth look into their feelings and actual behavior regarding authentication. Some of our high-level results include the failure rates for different kinds of authentication, and evidence of the diversity of people's authentication behavior and likes and dislikes. Further, we find that speed and ease of authentication are more important to many people than the failure rate (false reject rate). People's opinions on authentication vary greatly, with some considering it very little trouble and others finding it extremely annoying. People's favorite and least favorite kinds of authentication also vary greatly. Actual authentication behavior varies substantially across people in terms of the number of different targets they authenticate with, how well they secure them, how often they use them, and more. There's a surprisingly high average failure rate in authentication attempts (e.g., 3.3% for physical keys and 5.1% for passwords).

4.2 Related work

There is a tremendous amount of existing and ongoing work related to our project, especially in the areas of authentication, user studies, and wearables. We confine our descriptions of related work to examples of user authentication behavioral studies we perceive to be relevant to our particular study. We note that even definitions of “authentication devices” differ across studies, with some including the presentation of “things you have” such as keyfobs, and others only including tokens that display or contain information specific to a single individual, such as a badge with the owner’s photograph [91].

Several studies focus on smartphones and how people choose to secure them or not; their results vary considerably. A study of 1,003 Americans, age 18 and older, found that “people care [about privacy] but exhibit risky behavior”; about 56% of people surveyed did not employ any locking mechanism (e.g., a PIN or passcode) for their phone [50]. Other studies see fewer people choosing not to employing any locking mechanism. Egelman et al. report that 8 of their 28 interviewed participants (29%) and 42% of their 2,418-person online questionnaire respondents did not use any locking mechanism on their phones [24]. Bruggen et al. observed 35% out of 149 participants chose not to use any locking mechanism on their phone [97]. In our study, 4 of 26 participants (15%) did not use any locking mechanism for their phones; we observed participants choosing risky authentication behavior in terms of password management and password sharing, despite being aware of the consequences of their choice.

A National Institute of Standards and Technology (NIST) study involved 23 NIST employees ages 20 and above carrying a written diary in which they recorded a wealth of information about their authentication events [91]. This study is perhaps closest to ours in that it covers not just smartphones, passwords, or online authentication behavior, but

also a few other types of devices such as badges (although not the wide range of targets found in our study). Their participants recorded an average of 23 events a day, which is significantly lower than the 39 average of our participants. This may be because the NIST employees recorded only 10.0% of their authentication events were to smartphones, whereas our participants recorded 58.3%.

Various other non-smartphone studies and essays explore passwords and how users manage, choose, and forget them [20,28]. A study of the password habits of half a million users over a 3-month period used a component in Windows Live Toolbar on users' machines to record password strength, usage and frequency metrics [28]. The study found users choose weak passwords and use them across multiple sites and that 4.28% of Yahoo users forgot them during the study. We see an even higher percentage of users who forget, struggle to remember, or reset a password at least once during the study (36%). Hayashi and Hong conducted a diary study with twenty-one participants, in which participants carried diaries and recorded information about password-based authentications, and they found that participants performed more authentication at home than at work [32]; we see similar observation in our user study. A *New York Times* study explores the meaningful personal information users embed in their choice of passwords [96]. All of these studies agree with ours in concluding that users find it hard and frustrating to manage passwords according to established rules of safety. Usable security that takes into account human limitations and strengths has become increasingly important [21]. A recent study of online safety covers opinions and practices of both experts and non-experts regarding how to stay safe online [36].

We believe our study is unique in three ways. First, we enable the diary study with a wearable application to allow even easier and more streamlined in-the-moment logging of authentication events. Our motivation is to reduce under-reporting and provide more

accurate timing information for authentication events. Second, our study covers a wider range of types of authentication, including authentication with locked cars, doors, bicycles, public transportation and so forth. While there are studies that report on issues with specific kinds of authentication with physical targets, such as keyless cars [44], we are unaware of a study that covers the breadth of physical authentication targets accessed by the participants in our study. Third, our study provides quantitative and qualitative data about people's authentication behavior, unlike any other study to the best of our knowledge; using this rich dataset, we can analyze people's opinions about authentication in the context of their actual authentication behavior.

4.3 Study goals

Our user study was primarily an exploratory study; we did not have any specific hypothesis to test. We were curious to understand people's authentication behavior in general, but we specifically wanted to learn more about people's behaviors with regards to the following questions.

1. How often do people authenticate to physical objects (e.g., door, car)? How often do they authenticate to their computers?
2. What are the things (e.g., phone, door, ATM) people authenticate to and how often? What do they use for authentication (e.g., with password, tokens, fingerprint)?
3. How often do people encounter errors during authentication?
4. How do people manage their passwords? How often do people encounter errors in password-based authentication (e.g., because they mistyped or forgot)?

5. How does authentication behavior vary across age and gender?
6. How accurately do people guess the number of authentications they perform?
7. What is the authentication pattern for people? Do they perform more events on a weekday vs. a weekend? How are the authentication events distributed across a 24-hour period? Where do people perform their authentications (e.g., home vs. work)?
8. What are people's opinions about authentication? How do opinions vary with authentication behavior (e.g., opinions of people who authenticate more or encounter more failures)?
9. What do people like and dislike to use for authentication (e.g., fingerprint, password, or tokens)?
10. What physical objects do people carry with them for authentication?
11. What are people's thoughts on privacy in the context of authentication?

We discuss our findings for these questions in Section 4.7.

4.4 Methodology

We conducted a user study with twenty-six participants to understand their everyday authentication behavior. In this section we describe how we define an authentication event, our wearable digital diary method for self logging, and our study procedure.

4.4.1 Authentication event

We define an *authentication event* as one where an individual has to demonstrate, actively, that he is the right person to gain access to a resource or service through something he *is*,

something he *knows*, or something he *has*. Examples include unlocking the phone, unlocking the house door, logging in to a password-protected website, or entering a pincode on an ATM machine. Accessing a website with cached credentials that does not require even a mouse click requires no active user effort, so it does not count.

We define an *authentication target* as the device, resource, or service to which the individual requests access, and an *authenticator* as the evidence the individual provides to gain access. For example, when unlocking a phone the phone is the authentication target and the passcode is the authenticator; when opening a door with a badge, the door is the authentication target and the badge is the authenticator. Below is the list of targets and authenticators we used in our user study. Note that some of the items represent a category. For instance, “Laptop” also covers desktop computers, while “Password” also covers passcodes, pincodes, locker combinations or any knowledge-based authenticator.

Authentication Targets: Laptop (also desktops), Phone, Tablet (also e-readers), Website (also online websites or any software), Door, Car, ATM, Public Transport, Bicycle (also motorcycle), Phone payment, Card payment, Bank check, Locker (also locked drawers), and Other.

Authenticators: Password (also pincodes, locker combinations, etc.), Fingerprint, Face biometrics, Voice biometrics, Card (ID cards, credit cards, badges), Certificate (PKI), Mouseclick (where the participant has to click to authenticate, e.g., to request autofill with a password manager), Lockkey (physical key), Keyfob (remote key), Signature (physical ink), 2 Factor, and Other.

Depending on the number of required authenticators, authentication can be classified into three categories: *single-factor* (requires one type of authenticator), *two-factor* (requires two

types of authenticators), or *multi-factor* (requires more than two types of authenticators). To simplify logging of authentication events, we chose to focus on single-factor authentication and designed our interface for logging single-factor authentication. We asked participants to choose “2 Factor” as the authenticator when they perform a two-factor authentication, and if they perform an authentication using more than two factors or using an authenticator not listed in the above list, we asked them to choose “Other” as the authenticator.

We were only interested in the unlock events, i.e., the authentication events that lead to the individual getting access to a resource or service (e.g., unlocking a phone, unlocking a door); we were not interested lock events (e.g., locking a phone, locking a door). We were also interested in whether an authentication attempt succeeded and the location where it occurred. We asked our participants to log relative success using one of three provided icons: a yellow happy face to indicate success on the first attempt; a green unhappy face to indicate success that required more than one attempt; and a blue sad face to indicate failure. We wanted to collect semantic locations for authentication events, including Home, Work (includes School for student participants), Shop, Traveling, and Other. Thus, in our study an authentication event is represented as $\{event-time, authentication-target, authenticator, success, location-label\}$. In addition, participants could add comments to authentication events if they desired; we asked participants to comment on an authentication event if something out-of-ordinary happened.

4.4.2 Wearable digital diary

To avoid the under-reporting and poor recall that can affect diary studies [46], we wanted to enable immediate, easy logging of events. This is especially important for events such as authentication that can happen frequently and at times when it is inconvenient to pull



Figure 4.2: Easily accessible diary entry app on the smartwatch.

out a paper diary and pen or pull out a smartphone to bring up an app. We considered using a wearable voice recording device, but pilot study subjects said they would not be comfortable talking to themselves when unlocking stuff. We chose a smartwatch as our primary logging device, as it is easily accessible. Participants could access the logging UI in our app to log an event by simply raising their wrists; Figure 4.2 shows the watch UI that was available to participants when they raised their wrists. Indeed, most of our participants found logging events via the watch very convenient. We chose the commercially available Motorola MOTOACTV watch for our study [60]. We wrote an app for logging authentication events on the watch. Participants were given a MOTOACTV watch, with our app pre-installed, to wear for the duration of the study. We also developed a companion smartphone app, where they could view, edit, label and comment on their logged events using the bigger display.

Watch app

The MOTOACTV runs Android 2.3, but it is not programmable out of the box. To use it as a digital diary we rooted the watch and installed our Android application. Our app always ran in the foreground so that it was immediately accessible to participants, when they raised

their wrist or pressed the power button (Figure 4.2). The app also collected GPS locations and automatically synced with the companion smartphone app on the participant's phone every hour.

If logging is not quick and easy, participants are likely to delay logging and forget to do so later. We wanted participants to log four things about an authentication event: authentication target, authenticator, whether it was successful, and how many attempts they had to perform for authentication. Designing an easy logging interface for small screen device is challenging. After several iterations and feedback from pilot subjects we came up with a novel “slot machine” like interface to log an authentication event with as few as two taps on the watch touchscreen.

Figure 4.3 shows the application interface. The app screenshot on the left (Figure 4.3a) shows the main application interface, which contains three vertically scrollable columns of icons: the first for targets, the second for authenticators, and the third for success level; in the screenshot, phone and fingerprint are chosen as the authentication target and authenticator, respectively. The order of icons in each column was configurable for convenience, so participants could select their most-often used targets and authenticators without scrolling. (The watch had a touchscreen display, so participants selected an icon by tapping on it.) The app also cached the previously selected authenticator for each target, and would automatically select it when the participant selected the target, to reduce taps in the common case.

The third (emoticon) column was to log the number of attempts during authentication. The happy emoticon (top emoticon) was to log an authentication that was successful in the first attempt; the sad emoticon (third emoticon from top) was to log an authentication that failed, i.e., the participant could not authenticate. The unhappy emoticon (second emotion from top) was to log an authentication that was successful but required multiple attempts;

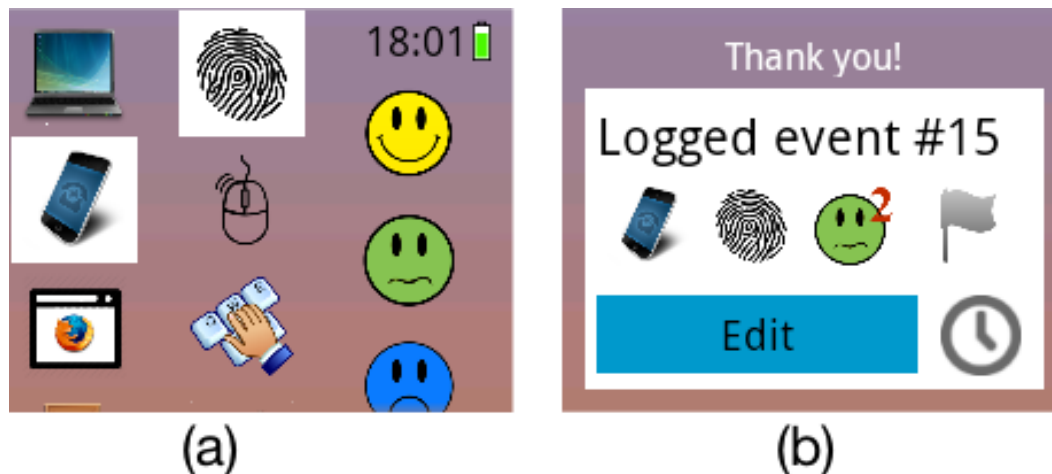


Figure 4.3: Watch app UI. a) Main watch app screen showing logging in progress for a phone event using a fingerprint unlock. b) Watch app screen confirming the logged event; the green icon with number two indicates that two retries were required to unlock the phone.

choosing the unhappy emoticon would bring a fourth column on the display (from the right side) with a vertically scrollable list of numbers (1, 2, 3, 4, and 5+) representing the total number of attempts the participant had to perform during authentication. Once the participant selected the authentication target, authenticator (if different than the auto-selected authenticator), and an emoticon (and the number of attempts if there were more than one) – all on the same app screen – the authentication event was logged and the participant was shown a confirmation screen with information about the logged event. Figure 4.3b shows a confirmation screen for an authentication to phone with a fingerprint; the authentication was successful but required two attempts (note the two superscript on the emoticon). The confirmation screen also provided the option to edit the event, in case the participant made a mistake when logging; adjust the event time (using the clock icon), in case the participant did not log the event immediately after s/he performed the authentication; or flag the event. We asked participants to flag events if there was something unusual about them or if they wanted to comment on them, which they could easily do when reviewing their event logs in the smartphone app described below. We inquired about flagged events and any other odd

events during participants' end-of-study interviews.

Smartphone app

The MOTOACTV watch allowed participants to log an authentication event quickly without needing to reach for their phones, but its small screen size made it difficult to view or edit their logs, so we used a companion smartphone app.

The smartphone app provided a dashboard interface for participants where they could sync the phone with their MOTOACTV watch, sync the app with the our server, browse and edit their authentication logs, and take the daily survey; the app ran in the background and synced with the watch and the server every 1 h and 6 h, respectively. Figure 4.4a shows the event log UI for a participant: each row represents an authentication event, with the time of event displayed on the left, followed by the authentication target icon, the authenticator icon, an authentication success/fail icon, a comment icon (indicating whether the participant entered any comment for the event), a flag icon, and a location label.

We asked participants to provide location schematic labels from a predefined list of five location labels: Home, Work, Shop, Travel, and Other. After a participant provided labels to a few events, the app automatically attached semantic location labels to other events using the location information, which it collected every 5 min. In the app screenshot (Figure 4.4a), the first and the last location labels were provided by the participant; the app inferred the other two location labels. After the study we deleted the location information (location coordinates) and kept only the semantic labels, as they are far less privacy-sensitive.

The app also automatically logged the participants' phone unlock attempts, both failed and successful; we used the Android Device Administration API and Android Device Administration feature to log failed and successful attempts. We could not automatically log

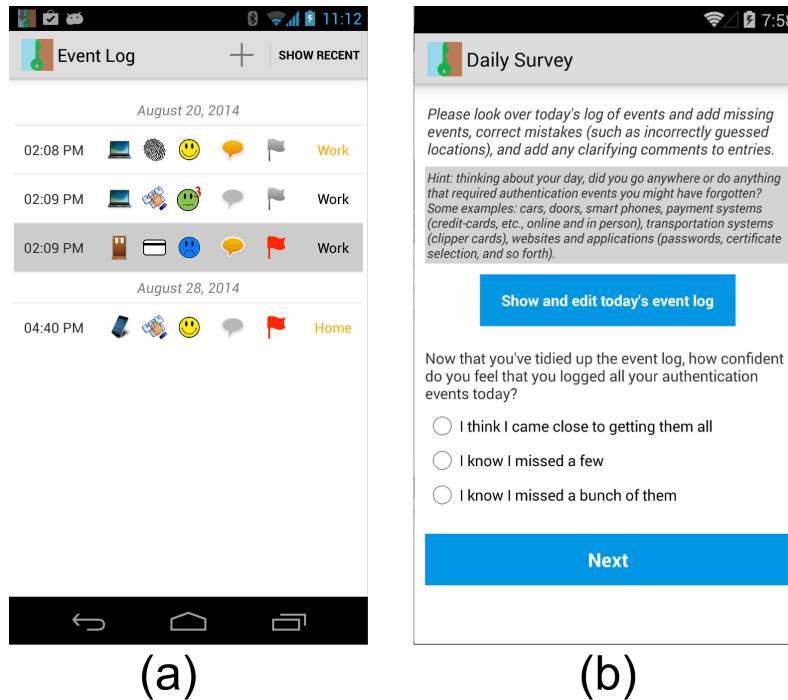


Figure 4.4: Phone app UI a) event log, b) end-of-day survey.

phone unlock events for participants who used an iPhone, so we asked them to manually log their phone unlock events on the watch.

We administered a daily survey at the participants' chosen time, usually in the evenings; Figure 4.4b shows the survey that participants received. In the survey, we asked participants to go over the day's authentication event log and make any edits if necessary, add comments to flagged events (if they wished), and add location labels to the authentication events. To simplify review of the event log, we only displayed events since the last time they took the survey; participants could see the complete log by choosing the 'show all events' option. In the survey, we also asked them how confident they were about their event logging – whether they logged all of the events (Figure 4.4b); at the end of the survey, participants had an option to provide a comment about the study or about their authentications on that day.

4.4.3 Study procedure

We performed a 3-person 2-day pilot study to test the digital diary for logging, for our categorization of the authentication targets and authenticators and to refine our logging UI. We then conducted the main study, which we describe below. The study was conducted at HP campus in Palo Alto, California, between June and September in 2014, during my internship at HP Labs. The study was approved by the ethics committee equivalent in HP.

Recruitment and enrollment

We recruited participants by word-of-mouth. We screened participants to verify that they were comfortable using a smartphone and were either affiliated with HP or U.S. citizens (a legal requirement from HP). We provided informed consent and information sheets to screened participants. If they agreed with the documents, we invited them to come in person to our lab (or meet via Skype for remote participants) where they signed the consent form and we interviewed them and explained the study procedure. Enrollments occurred throughout the week, and participants were asked to perform the study for 7 days from the enrollment day. We explained both in writing and in person what information we would collect. We also warned participants, both in writing and in person to practice safe logging, “Please only log events on the watch and phone when it is safe to do so. Please do not use the devices while driving, biking, crossing streets, operating heavy machinery, or anything else what would be risky!” We also informed them that they could withdraw from the study at any time for any reason or no reason; one person did so.

We gave each participant a MOTOACTV with our app pre-installed and asked them to wear it on their wrist; if they were uncomfortable wearing it on their wrist, they were allowed to attach it elsewhere via a provided clip. We also installed the companion smartphone app

on participants' smartphones; participants who did not have an Android smartphone received an Android phone with our app pre-installed, for the duration of the study, to take the daily evening survey and view the event log. Study participants received \$100 gift cards upon completion of the study.

Pre-logging interview

We conducted an in-person interview with each participant to learn about their own pre-study perspectives on their daily lives, the devices and resources they use, the authenticators they carry with them, and how they feel about various aspects of authentication. We asked participants to tell us about the authentication events they perform on their typical day by thinking about their typical day, going through their daily routine, and recollecting the authentications they perform. We also asked them to guess how often they might authenticate with various resources so that we could compare this information later with their actual recorded results. We used a set of questions to guide these semi-structure interviews, but we allowed the participant to digress and describe their opinions and behaviors about authentication in their daily life (see Appendix A.1 for the list of questions). We answered any questions they had about how to enter various kinds of events.

End-of-day survey

At the end of each study day, participants received a notification on their smartphones to take a survey. Participants could set the time when they would receive the notification. The survey asked the participants to review their daily event logs, add missing events, edit events (such as changing the authenticator or the event time), label event locations, and add comments if they had any. They also indicated their level of confidence in capturing events for the day

and could enter overall comments on the day if desired.

Post-logging interview

We conducted another semi-structure interviewed with each participant after the week of self-logging. We asked them about the logged entries we did not understand, about any authentication failures they logged, about their post-logging thoughts on authentication, the watch UI, and future inventions they would like to see in the area of authentication aids, about their choice of best and worst authenticators, and about how their authentication behavior might have changed during, or as a result of, the study (see Appendix A.2 for the list of questions).

4.5 User study participants

We recruited 26 participants to log their authentication events for one week each over the course of three months in 2014. Their ages ranged from 13 to 64; Figure 4.5 shows their age distribution. The participants included 8 females and 18 males; 16 (61.5%) were from Computer Science (CS) related fields, 9 (34.6%) were from non-technical fields, and 1 was from a medical-related field. Our participants represented 7 different schools, 4 different companies, and three different regions of the US. Participants self-reported their ethnicities as 14 Caucasian, 2 African American, 7 East Indian, and 3 Asian.

4.6 Study Limitations

We present our study limitations before our findings, so that the reader is aware of them before reading the findings in the next section.

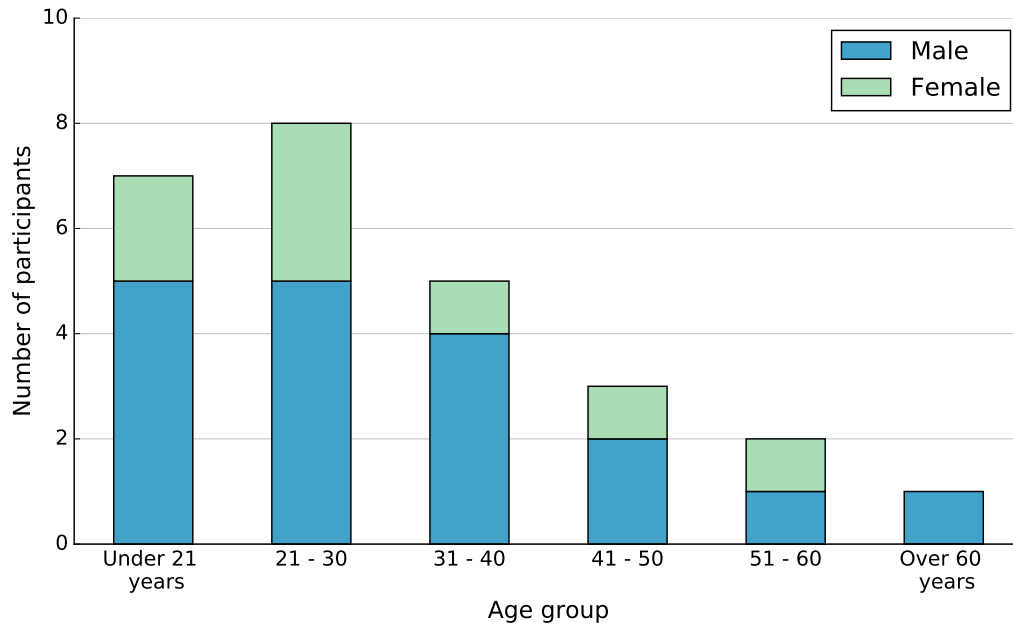


Figure 4.5: Study participant demographics, by age and gender (n = 26).

1. *Small sample size.* Regardless of their diversity, twenty-six participants is not a large enough group to give us good statistics about the overall population or any particular demographic.
2. *Participants with only daytime jobs.* None of our participants were night workers – they were all students or employees with daytime jobs, so we see only a few events at night.
3. *Mostly Android users.* Only five of our participants were iOS phone users with the rest being Android users. With more iOS users, for instance, we might have seen more fingerprint authentication.
4. *Limited location labels.* Our study did not support participants with more than one workplace as well as we would like; in our study there was only one label for workplace, *Work*. Participants with different work sites felt we should have provided multiple *Work* labels. In addition, we did not provide the breadth of semantic labels provided

by some other studies [41]; we chose to limit location labels to five labels to simplify the labeling task for participants.

5. *Missing location information.* GPS readings are not always reliable depending on location (for instance, indoors) and some participants' phones had trouble getting frequent GPS readings. Participants could assign location labels to events, and we encouraged them to do so by reminding them to label events during the daily evening survey. Participants did not assign labels to all unlabeled events. As a result, about 25% of locations in the study have no semantic labels attached.
6. *Extra phone unlock events.* There may be extra phone unlock events entered for participants unlocking their phones to take the daily survey. When queried, participants said they only logged these events if they also used their phones for another purpose, but we have no way of verifying this.
7. *Change in users' authentication behavior.* Users may have changed their authentication behavior as a result of participating in the study. One of our exit interview questions targeted this concern. One participant said he was more aware after the study and that his participation would lead him to change his opinion of the burden of authentication from a 2 to a 3 (see Section 4.7.8). Three participants told us they changed their behavior during the study in that they typed passwords more slowly to avoid errors, and one of them said he used his phone less often on some days, because he was embarrassed by how frequently he used his phone.

4.7 Findings

In our study participants self-logged their authentication events, and we conducted semi-structured interviews with all participants before and after the data-logging phase of the study. Together, participants performed 7,225 authentication events: they manually logged 3,488 authentication events, and our phone app automatically logged 3,737 phone unlock authentication events. The log for one of our participants did not cover quite the full week; for calculations where this could affect results we use only 25 participants. We took detailed notes from the semi-structured interviews. Our notes included direct quotes from participants and summaries and paraphrase of participants' explanations and descriptions about their authentication behavior and opinions. We followed the inductive and deductive coding methodology: once we had all the data from the interviews, we identified themes and categories in our notes (coded) and formed a data matrix, with columns as themes and rows as participants [56]. In this section we present descriptive statistics from the authentication events logged during the study, and the qualitative analysis from the interviews.

4.7.1 Digital vs. physical authentication

We were interested in learning how many authentication events are involved with digital versus physical targets. We classify authentication events as *digital* or *physical* loosely based on whether the authentication is to an electronic device/resource or to a physical object. We consider authentication as digital event if it is to a personal computer (at home or at work) or to an electronic resource authenticated through the personal computer (e.g., websites); an authentication is physical event if it is to objects in the infrastructure (e.g., door, ATM, authentication to use public transport). Specifically, in our study, we classify all authentication events where the authentication target is Phone, Laptop, Website, or Tablet as

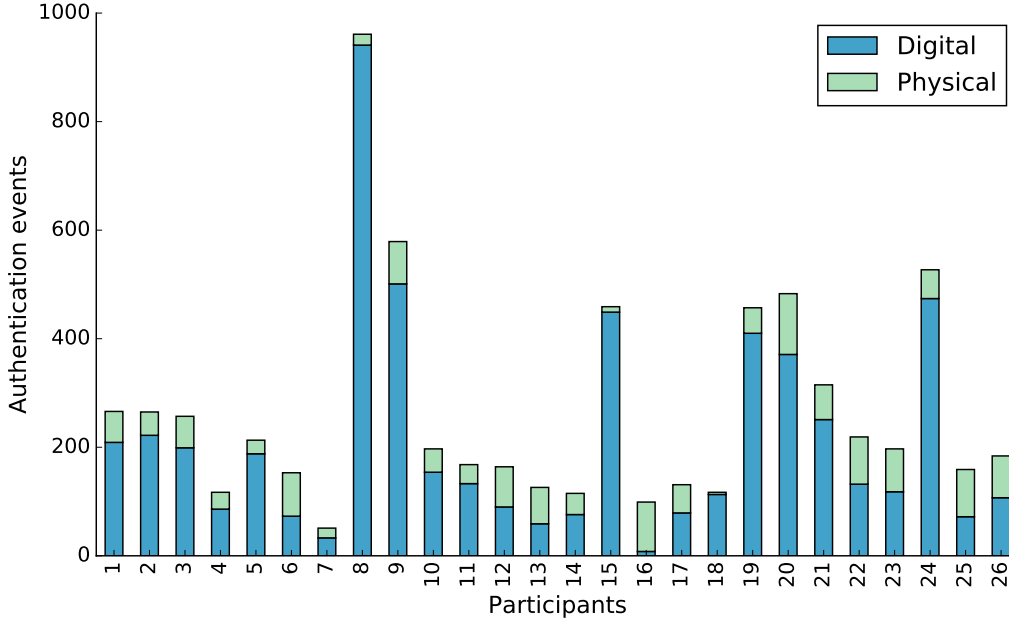


Figure 4.6: Authentication events logged by participants in the user study.

digital authentication events, and all other authentication events as physical authentication events. There are other ways to categorize authentication events (e.g., based on whether the authenticator is a physical object); we chose this categorization because we wanted to separate authentication events an individual performs on a personal computing device and authentication events performed outside a personal computing device.

In our study, physical authentication events were about 22% of all authentication events. Figure 4.6 shows the number of authentication events, digital and physical, each participant performed during the study. On average participants performed 8 ± 4 (mean \pm standard deviation; median = 8) physical authentication events per day, and about 31 ± 29 (median = 20) digital authentication events per day. For our participants, physical events accounted for about 9% to 40% (median = 22%) of their daily authentication events; most of these events were authentication to doors and cars. There is a high variance across participants, but it surprised us to see the substantial percentage of physical authentication events; indeed,

Table 4.1: Distribution of authentication events logged in the user study, by authentication targets; N is number of participants who authenticated to the target at least once. The first four targets are digital authentication targets.

Authentication targets	Events (%)	N
Phone	58.4	22
Laptop	12.6	26
Website	7.0	26
Tablet	1.5	10
Car	8.6	20
Door	7.7	24
Card payment	1.5	20
Locker	0.8	9
Other	0.7	11
Bicycle	0.6	7
Public transport	0.2	3
ATM	0.1	5
Check	0.1	5
ID verification	0.1	3
Phone payment	0.1	2

several participants complained about the frustrations with physical authentication events.

4.7.2 Authentication targets and authenticators

Table 4.1 shows the distribution of authentication events in the user study, by different authentication targets; the table also shows the number of participants who performed authentication to the target and used the authenticator, at least once. Four out of 26 participants did not use any security mechanism for their phone, so they did not need to authenticate to their phone. All participants performed authentication to a laptop (or desktop) or a website; most participants authenticated to a door; and many participants authenticated to a car or made a card payment (swiped the card in a card reader). The variations we see across participants is in part due to their widely varying ages: middle-schoolers, for instance, do not need to unlock cars as often as many adults, and most of them do not have credit cards. Most of our adult participants drove cars more often than they bicycled.

In our dataset, among all the phone unlock events, 92% occurred with passwords, 7.7% with fingerprint, and 0.3% with swipe gestures. For Laptop unlock events, most participants used passwords, but in some instances (0.8%) participants only had to click the login button because the laptop was set to auto-login. For Websites, which includes online services and access to software on phones or laptops, participants used passwords for 75.67% of all Website events and they used Mouseclick (auto-filling the password with a password manager or cached passwords) for only 21.3% events. We were surprised to see a low usage of password managers or cached passwords, because we expected participants to use password managers more or cache their passwords in the browser, as many said they did in their pre-logging interviews.

We see a lot of variance in the phone unlock behavior. Participants unlocked their phone about 23 ± 29 times per day (median = 13, min = 4, max = 133). The unlock usage we observed among our participants matches with the unlock usage of 25 ± 20 times per day reported by Hintze et al. [34] from their analysis of about 2,000 users' smartphone usage in the device analyzer dataset [98]. In another smartphone usage study ($n = 52$), Harbach et al. report a higher unlock usage, about 47 ± 26 unlocks per day [31]; we suspect the higher unlock usage is due to their participant demographics – a majority of their participants were young college students (median age = 23 years).

Table 4.2 shows the distribution of authentication events in the user study, by different authenticators; the table also shows the number of participants who performed authentication with the authenticator, at least once. Overall we find that 75% of authentication events involved secrets (Password), whereas token-based authentications (Card, LockKey, and Keyfob) accounted for 18.1% events. All participants used a physical key or a card, except our young participants: P10 (13 yr) did not use a physical key and P7 (14 yr) did not use a

Table 4.2: Distribution of authentication events logged in the user study, by authenticators; N is number of participants who used the authenticator, at least once.

Authenticators	Events (%)	N
Password	75.0	26
Lockkey	7.3	25
Card	6.1	25
Keyfob	4.7	18
Fingerprint	4.3	6
Mouseclick	1.6	15
Signature	0.4	13
Other	0.3	7
2-Factor	0.1	3
Certificate	0.1	4

card.

4.7.3 Authentication failures

We were interested to learn about the failures people encounter during authentication. We compute failure rate for an authenticator (or target) as the number of failed authentication attempts divided by the total number of authentication attempts with the authenticator (or to the target). Failed attempts is the number of times a participant attempts to authenticate to a resource and fails; for instance, if a participant successfully unlocks her phone in three attempts, she encountered two failed attempts. Note that these are all failure of the authentication method to verify the user (false rejects), and not the failure of the authentication method to deny an unauthorized user (false accepts), because self-reporting of events is unlikely to tell us if any of our participants were attempting to break into something they should not have.

Table 4.3 shows the failure rates for different authenticators and targets in the user study. The table shows two failure rates for each authenticator and target: FR across N is the mean (\pm standard deviation) across all participants who used the authenticator/target at least once,

Table 4.3: Failure rates for different authentication targets and authenticators; mean \pm standard deviation across N (all participants) and N_{10} (participants who performed ten or more authentications with the target or authenticator).

	FR (%) across N	FR (%) across N_{10}	N_{10}
Authenticators			
Card	2.2 ± 5.1	2.0 ± 3.5	17
Fingerprint	25.6 ± 24.7	13.4 ± 20.6	4
Keyfob	0.6 ± 1.7	0.8 ± 1.9	13
Lock key	3.3 ± 7.8	0.7 ± 1.6	18
Mouseclick	5.6 ± 10.7	2.6 ± 3.7	2
Password	5.1 ± 5.7	4.8 ± 5.6	25
Targets			
Car	0.4 ± 1.4	0.5 ± 1.6	16
Card payment	1.5 ± 4.8	7.3 ± 9.5	4
Door	6.2 ± 14.7	2.0 ± 4.2	18
Laptop	7.7 ± 8.7	8.4 ± 9.1	21
Phone	4.4 ± 9.1	4.4 ± 9.1	22
Tablet	0.3 ± 0.9	1.0 ± 1.8	3
Website	10.4 ± 15.1	7.8 ± 5.6	11

and FR across N_{10} is the mean (\pm standard deviation) across all participants who used the authenticator/target ten or more times. Some participants in the study used a target or authenticator just a few times; they do not provide enough authentication samples to measure the failure rate for that target or authenticator. So we also computed the average failure rate considering only the participants that provide enough (ten or more) authentication events for a given target/authenticator during the user study.

We observed a higher percentage of authentication failures than we expected. The average failure rate for Laptop, Phone, and Website was 8.4%, 4.4%, and 7.8%, respectively. The average failure rate for passwords was about 4.8%; surprisingly, even mouse clicks were problematic for some participants, with a 2.6% failure rate. Mouseclick refers to an authentication event in which participants used a mouse click to authenticate (e.g., choosing a certificate, auto-filling a password entry); the failures in mouse click authentication are

instances when the user accidentally chose the wrong certificate or accidentally auto-filled the wrong username and password (e.g., for websites where s/he has multiple accounts). Some participants struggled with physical keys; failures with physical key authentication were due to a participant selecting the wrong key from his/her key bunch and trying it on the lock.

The biometric fingerprint failure rate was high (13.4%), because among the six participants who used fingerprint for authentication, one participant (P18) injured the finger he uses for fingerprint authentication and as a result, suffered many failures (about 44%). The participant reported that he could not authenticate with the injured finger; he would retry until his phone allowed before falling back to the PIN-based authentication. If we exclude the injured participant, we see a fingerprint biometric failure rate of about 4.6%, which is still higher than we expected.

Overall, we see high variability in the failure rates among participants; Figure 4.7 and Figure 4.8 show the high variability in the failure rates encountered by participants (N_{10}) for different targets and authenticators, respectively. Majority of our participants did not encounter any failure when authenticating to a tablet, door, or a car. The median failure rate for laptop, phone, and website was 4.6%, 1.7% and 8.3%, respectively; most of the failures for Laptop and Website were with passwords, about 96% and 86%, respectively. Card, physical key, and keyfobs worked perfectly for majority of our participants (zero failure rate), except for some participants; five participants logged failures using a card to access a door, paying for public transport, or making a purchase (median failure rate = 6.6%). The median failure rate for password, fingerprint, and mouseclick was 3.7%, 4.6%, and 2.6%, respectively. The variability in password and mouseclick failure rates is less compared to other authenticators, which suggests most people encounter failures with these authenticators.

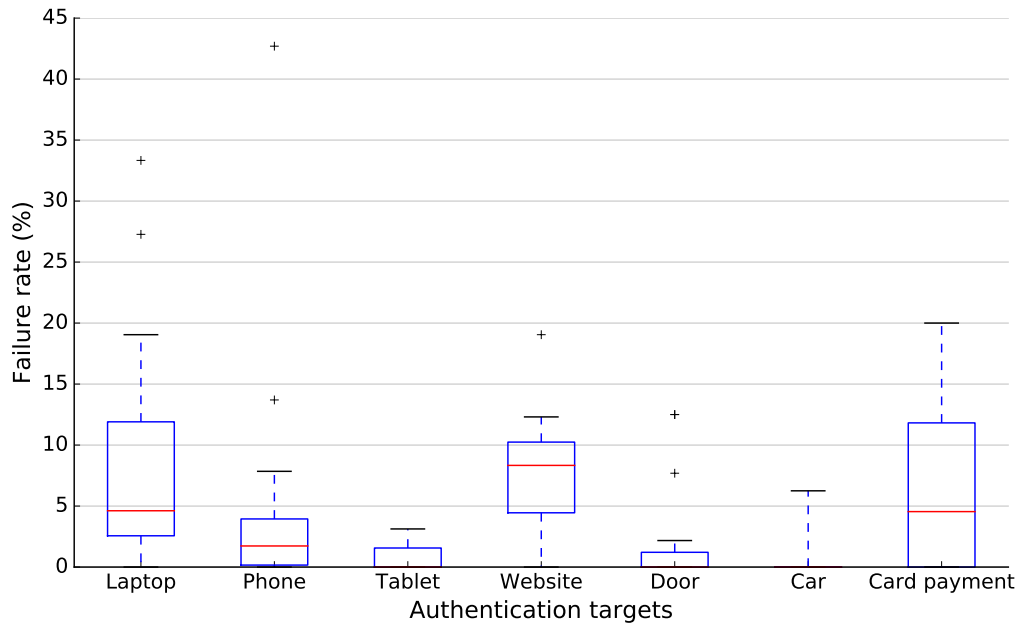


Figure 4.7: Failure rate for different authentication targets; including only participants who authenticated to the target ten or more times.

4.7.4 Passwords

Users' frustration with passwords are covered to a great extent in the literature; what we found mostly supports previous findings. The common complaints about the passwords, among our participants, were their length (which makes it more likely to make a mistake while typing them), need to memorize them, and having to type them frequently (on laptops and desktops). In general, participants liked that they can easily cache passwords on their browsers; two participants even said passwords were their most liked authenticators, especially the 4-digit passcodes on their phones, because they are easy and quick to type.

The password failure rate shown in Table 4.3 is a collective failure rate for all targets combined (e.g., laptop, phone, website, tablet). When we take a closer look at the password failures for individual targets, we see significant difference in their failure rates, as shown

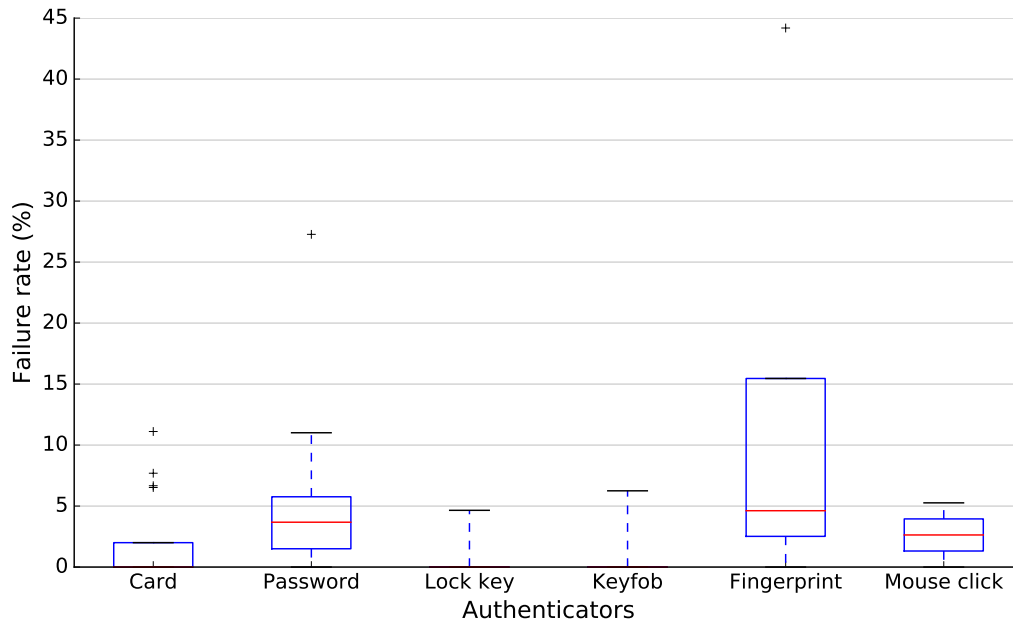


Figure 4.8: Failure rate for different authenticators; including only participants who used the authenticator ten or more times.

in Figure 4.9. Again, we consider only participants who performed ten or more authentications to Laptop, Phone, Tablet, and Website (N_{10} participants). For the majority of these participants, authentication to Website was most error prone (median failure rate = 6.7%), followed by Laptop (median = 4.7%), Phone (median = 1.4%), and Tablet (zero median failure rate; two of the three participants did not encounter any failure using password on their Tablet). It is interesting to note, except for Tablet, authentication targets with high median failure rate were used less: password-based authentication events for Website, Laptop, and Phone are about 5%, 12%, and 54%, respectively.

It was surprising to see a high failure rate for Laptop, because laptop or desktop passwords are frequently used, often typed several times in a day. In our exit interview we asked participants about their high failure rate with passwords. Several participants commented on making mistakes typing passwords (because of the length and/or complexity of the passwords) and forgetting a password, especially for websites that are not often used.

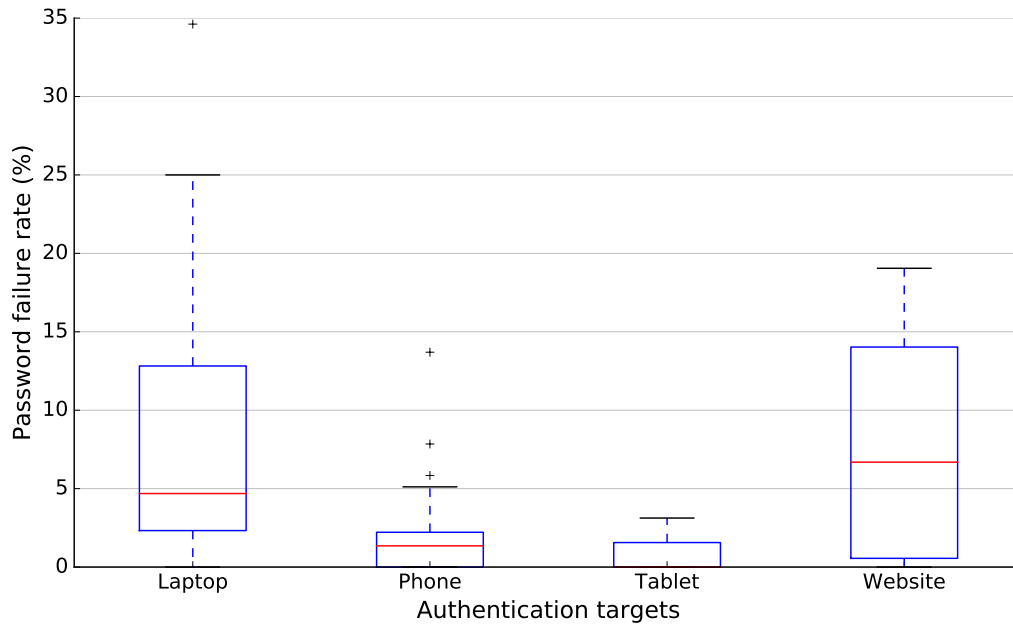


Figure 4.9: Password failure rates for different authentication targets; including only participants who authenticated to the target ten or more times.

This observation matches with past findings by Adams and Sasse that users have trouble recalling infrequently used passwords [2]. Several participants quoted “typing too fast” for getting passwords wrong, either out of habit or because they do not want anyone to see their password.

“I always type it super fast and get it wrong a couple of times.” (P16)

“It can be stolen easily, that’s why I’m always in a hurry in typing a password – it’s a mental thing, even if no one is around. It makes me type it quickly – it’s instinct.” (P14)

Typing an old password (due to muscle memory) when the laptop password was recently changed or getting confused between laptops and desktops they own/use and typing the password of one device on other, were two other reasons participants gave for making mistakes entering passwords.

“It’s muscle memory and I usually mess up when I update a password. I’ll type old ones first and of course it fails.” (P2)

“I’m on autopilot typing my password, which is different on my PC versus my Mac, so I have a ordered search of passwords I go through until one works.” (P1)

“I get them [desktop and laptop] mixed up, and I type the wrong password – it’s muscle memory.” (P3)

One participant commented on not being able to see a password when it is being typed, especially for long passwords. Another participant (P1) complained about his phone being less responsive if he ran too many apps in background, and “typing in the numbers [passcode] registered too slowly”, so he had to retry.

Many people feel that the rules around choosing and managing passwords have become onerous, especially in corporate environments. Among our participants who were employees, we found *no one* followed all our usual workplace rules for passwords: change them frequently, don’t reuse them, choose passwords of significant complexity, do not use the same password across multiple sites or accounts, do not write them down, do not cache them in browsers, and do not use a third-party password manager to store them.

“It’s awful. I’m dying...Everybody’s got different rules and people are requiring I change them and then I can’t remember them. Then life is hell...I use the same one [password] – I’m not a fool...All the tools to do my job are impossible to get to...This requirement that I change the password – They’re causing us not to be able to remember the password, not to pick a good one, to use the same one and just change the postfix, or to write it down. They’re forcing me into this

corner – I don't know what to do. Maybe I'll write it on a sticky note and paste it on my computer.” (P17)

These participants “cheated” in at least some regard – and they were aware of it. Immediately after they told us about a bad practice, they confessed that it was bad practice or justified their action.

“About the management aspect – remembering a password – I reuse passwords is how I get around it, which is bad.” (P2)

These comments may indicate that password management has become difficult enough that even tech-savvy employees are not willing to abide by the requirements.

To manage their many passwords, participants turned to a variety of tricks and tools: password-managers (n=9; 34%); reuse same password with permutations (n=8; 30%); reuse same password (n=5; 19%); save passwords in an encrypted file (n=5; 19%); save passwords in a plaintext file (n=3; 12%); cache passwords in browsers (n=5; 19%); write passwords on a physical paper and keep it hidden (n=1; 4%); or memorize (n=10; 38%). Several participants used more than one strategy. In a user study by Ion et al. [36], 19% non-expert users reused passwords, which exactly matches our results. We expected more participants in our study would use password managers, but only 34% participants did, which is higher than the 24% for non-expert users but much lower than the 74% for expert users in Ion et al. [36] and 81% users in a study by Stobert et al. [92]. We suspect the low percentage of password manager use in our study is because many participants' organizations were not benign toward password managers. One participant mentioned that being able to share passwords was important for him, and that was one of the reasons he did not use password managers.

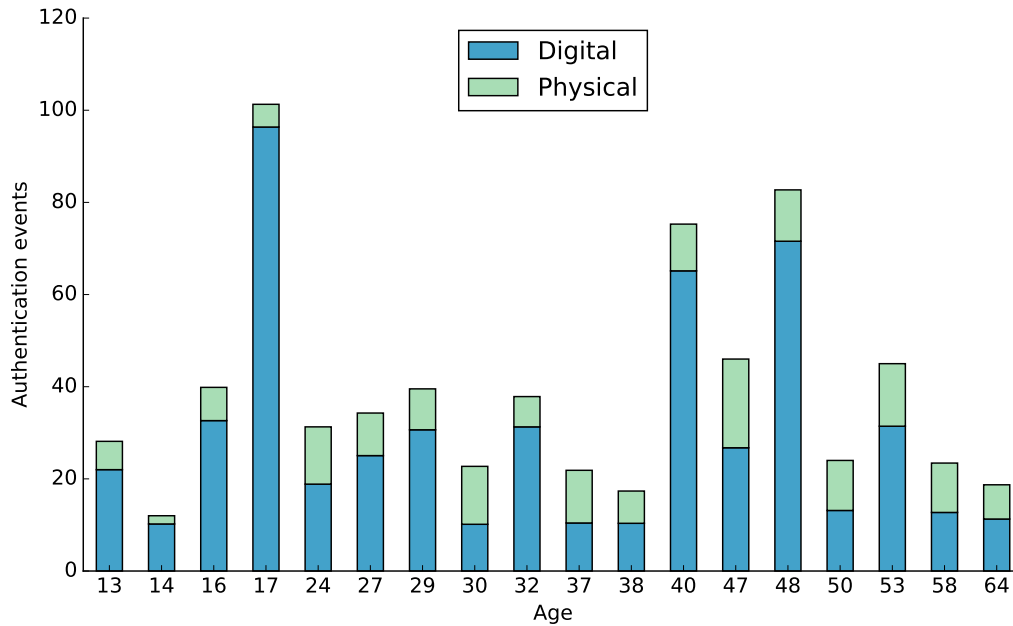


Figure 4.10: Average authentication events per day across age of participants in the user study.

4.7.5 Authentication behavior across age and gender

Figure 4.10 shows the authentication events logged by participants of different ages; for two (or more) participants with the same age, we show the average number of authentication events. We see a high number of authentication events in younger and older participants compared to the participants in their twenties and thirties. We believe there are no general conclusions to draw from this and that it is likely due to individual behaviors. We do not have enough participants of different ages to confirm the difference we see in our participants. We can, however, conclude that no one escapes authentication – even the youngest and the oldest participants perform about 20 authentication events, and everyone performs some physical authentication events.

We compared the authentication behavior between male ($n=18$) and female ($n=8$) participants. Both groups logged roughly the same number of authentication events, phone unlock events, physical events, and failures per day. The average authentication events per

day logged by the male group and the female group were 37 ± 33 and 42 ± 16 events, respectively. The average number of phone unlock events per day logged by the male group and the female group were 22 ± 33 and 25 ± 23 events, respectively. Finally, the average number of failures in a day logged by the male group and the female group were 2 ± 2 and 2 ± 1 failed attempts, respectively. The high standard deviation is due to the high diversity and variance in the study participants' authentication behavior. Overall, at least in our small sample size, we do not observe any notable different authentication behavior across gender.

4.7.6 Guessing authentication events

We were curious to see whether individuals are good at guessing the number of authentications they perform in a day. In the pre-logging interview, we asked participants how many times they believed they authenticated themselves every day on average. Among the sixteen participants who answered with a specific number, seven participants significantly overestimated their actual daily authentications. Nine out of sixteen participants, however, underestimated their daily authentications. Combined, the average of guessed daily authentication events was close to the actual number (44 ± 34 vs. 39 ± 28).

4.7.7 Authentication patterns

We also captured when, how often, and where participants authenticated themselves. Overall there was a high variance among participants. Authentication events per hour across participants in a 24 h day range from 0 to 14 with an average of 2. Authentication events per day across participants range from 7 to 137 with an average of 39 per day.

We were interested in learning whether participants log more events on weekdays than on weekends. Figure 4.11, which shows the average number of authentication events each

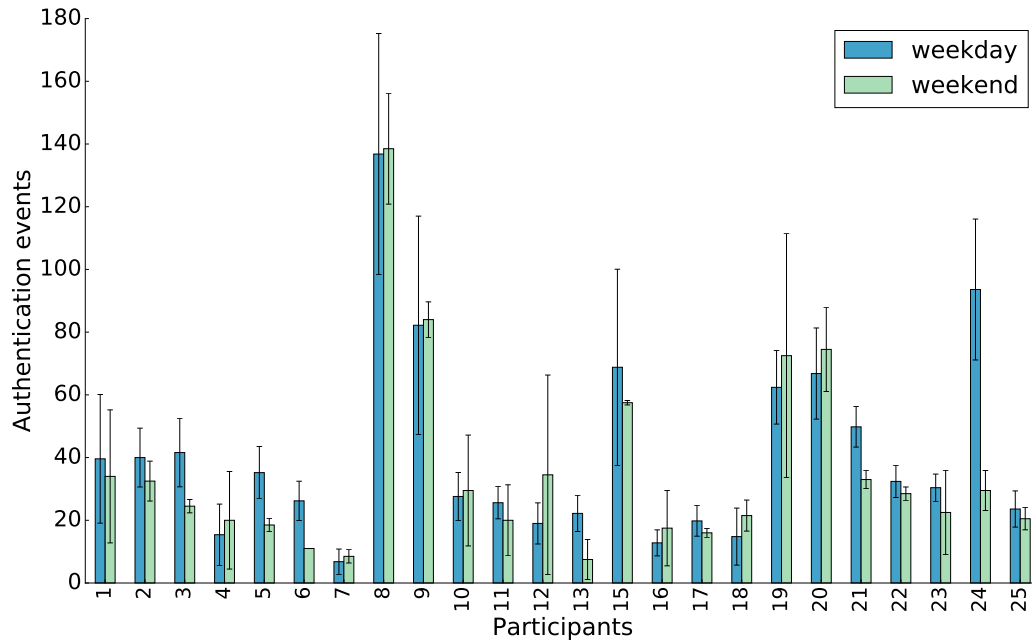


Figure 4.11: Average number of authentication events participants performed on a weekday (average across 5 weekdays) and a weekend (average across 2 days on a weekend); error bars show standard deviation.

participant logged during weekdays and on the weekend; we ignore P14 and P26, because they did not log during both days on a weekend. We see no obvious pattern across all participants. Only five participants (P3, P5, P6, P21, and P24; one male, four female) performed significantly more authentication events during a weekend than on a weekday ($p < 0.05$).

Figure 4.12 shows the number of authentication events participants performed at different hours of day during their one-week study. All our participants had day jobs or generally followed a day-oriented schedule, and this is evident from the figure, as there are few events after midnight. Nevertheless, there seems to be no hour of the day where someone was not authenticating with something.

Table 4.4 shows the number of authentication events performed at different locations. We expected to see most events occurring at work (where school counts as work for students),

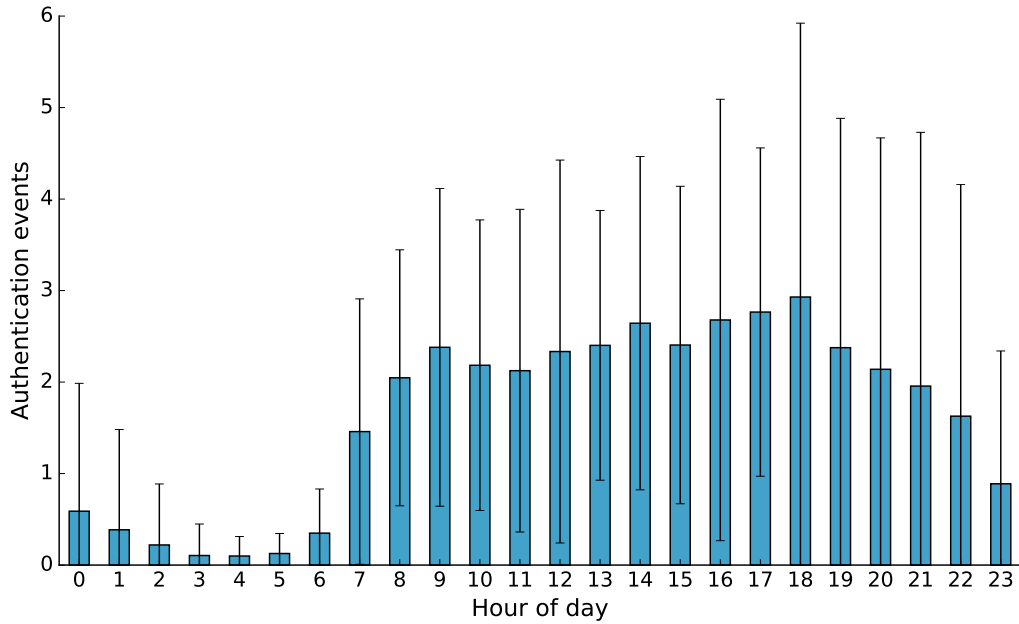


Figure 4.12: Average number of authentication events performed in one day; mean \pm standard deviation across all participants and all days in the user study.

Table 4.4: Fraction of all authentication events and just phone authentication events performed at different locations.

	All	Phone
Home	48.90%	59.83%
Work	38.26%	29.75%
Shop	5.04%	4.65%
Travel	3.92%	2.81%
Other	3.88%	2.96%

but we were wrong. Home is the location receiving the largest number of authentications.

There were roughly 10% fewer authentications on average at work, with shopping (which includes restaurants), traveling and other receiving far fewer events. It was interesting, but not surprising, to see that participants unlocked their phones 59.8% of the time when they were home and about 29.7% when they were at work.

Table 4.5: Participants opinion regarding authentication events on normal days (Normal) and on days when they encounter a failure (Failure); $n = 16$.

Opinion about Authentication	Normal	Failure
(1) I don't even notice them.	0	0
(2) I notice them, but they rarely bug me.	4	3
(3) They bug me, but not too much.	11	6
(4) They bug me and I'd like to avoid them.	1	3
(5) They are extremely frustrating.	0	4

4.7.8 Opinion about authentication

Our participants' opinions on authentication vary widely. In the exit interview, we asked participants to rate their overall feelings about authentication on a scale from 1 to 5, with 5 being extremely frustrating; Table 4.5 shows the opinion of the sixteen participants who gave a rating. Among the sixteen participants, twelve participants (75%) found authentication somewhat burdensome (Table 4.5; opinion 3, 4 and 5 on normal days). Six participants, unprompted, gave two opinions when asked about how they feel about authentication: first for how they feel in the normal course of things, and second for how they feel when something goes wrong, such as forgetting a password, losing a key, or having to change a password.

“They bug me a little [rating 3] but they give me a sense of security. Shoots to a 5 when I have to set up an account or service or use the phone to enter 15 character password.” (P20)

“Most of the time it's just the cost of doing business [rating 2] – until it breaks. Then it's a 5 because it stops me doing what I need to do right now.” (P12)

We analyzed participants' authentication log data to find correlation between participants' authentication opinions and the number of authentication events they performed or the failure rate they encountered. We expected participants who logged more events or who encountered more failures would be more frustrated (offer higher rating in the table), but we did not notice

any consistent correlation; there was a lot of variance among participants' opinions and their logged authentication events and failure rates. Interestingly, the participant (P9) who logged the maximum average authentication events per day (about 82) gave a lower rating of 2 ('authentications do not bug me'), whereas the participant (P13) who logged the minimum events per day (about 18) gave a higher rating of 4 ('authentications bug me and I would like to avoid them'); the participant (P15) with the maximum failure rate in a day (about 8%) gave a rating of 2, whereas the participant (P14) who had zero failures throughout the study gave a rating of 3. It could be the case that people who find authentication annoying take measures to minimize the number of times they have to authenticate, and people who are not annoyed with authentication do not mind performing it more.

The average opinion rating of female ($n=4$) and male ($n=12$) participants was 2.75 ± 0.5 and 2.66 ± 0.65 , respectively. We further explored how authentication opinion ratings differ across participants' age. We divided participants in two groups: teenagers and middle-aged adults (ages under 20 or 40 and above; $n=7$), and young adults (ages 20-39; $n=9$). Teenager and middle-aged adult participants on average reported less frustration (2.2 ± 0.4) than young adult participants (3.0 ± 0.5). Although the participants in our study were from a diverse background, the sample size is too small to generalize these results or establish any correlation.

4.7.9 Best and worst authenticators

The kinds of authenticators participants most liked or disliked varied greatly, as seen in Table 4.6 and Table 4.7; flash-to-pass (Table 4.6) is one participant's authentication method that allows her to open her garage door by flashing her headlights, which then also unlocks her house from the garage entry. Some participants' favorite authenticators were other

Table 4.6: Participants response for “Which authenticator do you like the most?”

Authenticator	Participants
Keyfobs	19%
4-digit pincodes	15%
Badges and passes	15%
Fingerprints	12%
Passwords	8%
Physical Keys	8%
Cached passwords	4%
Flash-to-pass	4%
I hate them all	15%

participants’ least favorite. Note that participants’ answers were their own and not chosen from a predetermined list. (If they had been from a predetermined list, we might have seen more people choose authenticators such as “cached passwords” as most-liked.) While we suspected many people would complain about passwords, we were surprised by the number who disliked physical authenticators such as keys and badges. Participants also sometimes distinguished between the number of times they had to use a particular authenticator versus the amount of effort required each time.

Although the most disliked authenticator varied among participants, for most participants *having to remember* something (either to have a physical token or a password) was the explanation. Another reason to dislike an authenticator (keys, especially) was the need to carry it.

“I don’t like my badge. I never remember to have it on me when I should.” (P21)

“I don’t like to carry around physical keys. It’s just another thing to manage.”

(P7)

“I like key fob as opposed to physical stuff. Remote authentication without physical contact is a much better experience than physical contact or swiping.

But the fob is too big – it’s difficult to carry.” (P13)

Table 4.7: Participants response to “Which authenticator do you dislike the most?”

Authenticator	Participants
Passwords	50%
Physical keys	27%
Badges and passes	8%
Barcodes	4%
Credit cards	4%
Everything	4%
I like them all	4%

Most participants liked an authenticator because it was either *automatic* (keyfobs or badges) or *quick* (4-digit PIN, fingerprint). Interestingly, participants who liked fingerprints said they encounter failures with fingerprint often – indeed, we saw that in their logs – but they did not seem to mind the failures, because it was quick to try again.

“The fingerprint reader sometimes fails because you have to have your finger aligned just right, but it’s fast to swipe it again.” (P21)

“The fingerprint swipe for my phone [is my favorite]. It failed a lot but you don’t have to do much.” (P18)

This seems to suggest that an authenticator being quick and easy is more important, at least to some participants, than it being accurate in terms of false rejects.

4.7.10 Things participants carry

While many problems with passwords are well-documented, physical authenticators also offer challenges for users. Some of us have many resources we need to access frequently using physical authenticators, and this means we need to carry many authenticators with us. To find out more about this potential problem, we asked participants if they were willing to dump out the contents of their wallets, pockets, purses, bags, or other places where they carry authentication material. We told them we did not need to see what they dumped out,

Table 4.8: The number of authenticators carried by participants, summed across all participants.

Authenticator	Count	Remarks
Bike key	4	Others used combinations
Cabinet/drawer type key	2	
Car fob	12	Regular or electric
Car key	17	
Car proof of insurance card	7	Others kept these in the car
Credit card	50	
Debit card	17	4 for work, 3 not used
Digital key	3	
Driver's license	16	
Health insurance card	9	
House alarm fob	1	1 key to amenities
House/apartment type key	19	
ID badge	14	Mostly corporate, also gym
Jewelry box key	1	
Key of unknown function	6	Others used combinations
Locker key	2	
Loyalty/gift/discount card	38	
Mail box key	5	
Membership card	17	
Motorola skip	1	
Other door key	17	2 expired
Other ID card	16	
Phone	2	Phone app for passwords
Public/private transportation	10	
Scraps of paper with writing	5	Zip card/buses/BART/metro
Total	291	

but that they could just enumerate for us what they found. Table 4.8 shows the results, added across participants.

There are several indicators that managing these carried authenticators can be difficult.

“I don’t like to carry around physical keys. It’s just another thing to manage, and if I were to ever forget it...The Pebble is one exception because it’s always on your wrist. If it had a computer unlock I’d be totally happy.” (P7)

Several participants attempted to divide up or stage their authentication material so that they did not need to carry all of it. For instance, one participant has bags for different purposes,

with the appropriate ID cards or badges in the different bags. Another participant uses a phone cover with slots for cards in it. He carries his driver's license, a debit card, and his badge in the cover. The rest of his cards he puts in his wallet, which he keeps in his car and only carries if he needs it in a store. Another participant stages his keys so that he carries a minimum but the keys he does carry allow him access to the rest of the authenticators.

“I’m at the limit of physical keys I can carry – can’t tolerate any more. It’s a layer system – the rest are kept in a pie tin at home. It’s part of the family semaphoring system. Know who is doing what where...I have it set up usually so most things are automated and I don’t have to carry as much. Never be without a house key – I teach all my kids that too.” (P12)

Another participant rigged up his own “smartwatch” in the form of a Motorola skip¹ clipped to his regular analog watch. He unlocks his phone by tapping it against the skip on his watch. Attaching it to the watch means he does not need to worry about carrying it – it is always with him since he wears his watch every day.

Another indicator is that people cannot track what they carry. They carry authenticators with them that they no longer need or could identify. People carried expired school IDs, unused credit cards, and keys whose functions they couldn't remember. For instance, one carried two unidentified keys and said

“But I’m scared to remove them. They seem like they might have been important.”
(P21)

They also can't find material they were sure they were carrying.

¹Motorola skip is a small NFC tag, available as a clip-on wearable and as a sticker, that can be used to unlock Motorola's Moto X smartphones. Users tap their Moto X on the tag to unlock the phone.

“There should be two health cards – one for kids – but I can’t see where that went.” (P26)

Some participants also make arrangements to carry authenticators on behalf of others. One teenager (P7) carries his brother’s gym ID card “‘cause he doesn’t carry a wallet. We go together and my parents are worried he’d lose it.” Another carries his own locker key and his friend’s. Another carries his friend’s house key, and a third carries her father’s house key too.

A couple of participants volunteered that it’s not just the hassle of carrying so much material that is the problem, but it’s also their mental anxiety over wondering if they might have forgotten something. These people wanted someone or some tool to help them manage their keys and cards.

“From a technological point of view – [I want] someone [to] tell me your key is this place or your credential info is here...[It would help] best at home – [I] put my keys somewhere – depends on situation – baby crying, sofa, piano, and then I forget [where I put them]. But when I try to use car first have to find key or I can’t use my car. So [if I could] have it be ‘go to the car and someone gives me this key’ that would be great!” (P13)

“Did I forget something? Constant confusion if I forget something.” (P8)

Changes to routine also increase the chance that people will not have the authenticators they need with them. One participant mentioned

“Traveling has a problem with acquiring more keys and cards...” (P12)

Emergencies are a further problem: during a recent fire drill at a participant’s company where emergency communications required particular tablets,

“The emergency crew didn’t remember to bring the tablets with them when exiting the building, or they had them outside in their locked cars but didn’t bring out the car keys.” (P12)

Finally, people carry scraps of paper in their wallets and bags with authentication material, sometimes obfuscated and sometimes not. For instance, one participant carries a paper with last year’s gym locker combo on it *“’cause I was constantly forgetting it and asking the coach to open it for me.” (P8)*

Another participant carries a paper in his wallet that is *“a letter of love to my wife – but it happens to be passwords encoded.” (P11)*

4.7.11 Privacy

One question many authentication studies consider is how much people care about privacy. We observed a higher level of care than we expected from our participants, with only two of the seven teenagers leaving their phones unlocked and two of the adults doing so. While our study does not include enough participants to make broad generalizations, we see evidence that teenagers and not just adults are interested in privacy and security, although teens may have less useful understandings of how to achieve it. We asked all participants why they chose to lock or leave unlocked their personal devices and resources. Both of the teenagers who did not lock their phones said it was because their phones always remained under their physical control, or in a safe environment (a desk at home). One of them also said he was careful not to keep anything private on his phone, and that he backed it up so nothing would be lost if his phone were lost. The other five teenagers all locked their personal devices with the intent of keeping them safe from the prying eyes of friends and sometimes parents and siblings.

“[I lock my phone] so people don’t just go inside my phone – it’s not pleasant for anyone this kind of snooping.” (P8)

Three of the teenagers and seven of the adults also mentioned that besides having activity timeouts on their personal devices that automatically lock them, they deliberately lock their devices whenever they put them down or walk away from them, regardless of timeouts.

On the other hand, both teenage girls (but none of the teenage boys) mentioned that they share their phone passwords with selected friends. This sharing seems to have social significance, and one of the teenagers suggested at the end of her exit interview that any kind of new authentication technology needs to support sharing of access.

“I want to use thumbprints on everything but I can’t pass thumbprints to others – some friends can have access to my phone but not everyone.” (P19)

4.8 Discussion

In this section we reflect on things we learned from the study.

4.8.1 What would we do differently?

A study of this sort takes a tremendous amount of time. If we perform a similar study in the future, we will try to enlist more than two people to help administer the study, support the study, interview participants, and analyze the data. We would also try to capture, through web surveys, information from a larger population, even if we could not get detailed log data from them. It would be interesting to compare opinions from a survey with those gained from participants in a detailed study such as this. In fact, we hope this will be part of our future work.

We learned that a pilot study is essential. Our study would have failed entirely without the changes we made as a result of the pilot study. In the future we might perform a couple of pilot studies to make sure the changes from the first pilot study perform as hoped in practice. We were merely lucky that they did.

Finally, we found that working with GPS data was particularly tricky. Participants provided semantic labels for log entries, and we used their labels and a parameterized region around the coordinates to guess the semantic labels of further entries. It took a lot of trial and error to determine how much variation in coordinates could go under the same semantic label and required our consulting with participants periodically. Next time we might ask users to label regions on maps that they consider falling under a particular label. The lack of uniformity in the ability of participants' devices to pick up location readings also added complication and error to our results.

4.8.2 What went well?

One of the reasons we created the phone app as well as the wearable app is that we worried many participants might prefer to log entries from their phones. After all, many people have their phones handy most of the time. We gave participants the choice of logging either from phone or watch. However, the immediate accessibility of the wrist wearable combined with our slot-machine style logging interface worked as intended. Except for three people, participants logged an average of 93% of their events on their watches. One participant (P25) did not like wearing a watch so he logged all events from his phone, and two other participants (P15 and P16) did not wear the watch because they thought it was not fashionable, but they carried it clipped to their bags and logged about 40% of their events on the watch. Overall, the approach made logging easy enough that 84% of events in our study

were logged from the wrist wearable despite the availability of the phone application. We suspect that this approach could lead to future wearable digital diary studies. This suggests that the smartwatch was the preferred platform for logging in-the-moment events compared to smartphone among our participants.

4.9 Summary

We learned a great deal about people’s authentication behavior from our user study; although our participant sample size was small, the participants came from diverse backgrounds, and the combination of qualitative and quantitative data from the study provided rich information. Our sample size was not big enough to draw statistically significant results, but we gained knowledge about people’s authentication behavior in the context of the questions we posed at the beginning of the study (Section 4.3). In our study, we learned that the number of authentication events to physical objects (cars, doors) was significant (about 22%); a majority of the authentications were to phone, laptop, and websites (in that order), and about 75% of authentications were performed with passwords (or PIN); token-based authentication (e.g., key, badge, card) worked perfectly for majority of our participants, but some participants encountered enough failures that they expressed their frustration with physical authentications. Password failure rates differ across targets (laptop, websites, phone), with high failure rates (5% to 7%) for laptop and websites. We observed that participants’ authentication patterns were fairly consistent throughout a week; authentication behavior seems to vary depending on age, but we did not see any variance based on gender. Participants liked automatic (e.g., keyfob) or quick (e.g., fingerprint) authenticators; they disliked authenticators that required them to remember or carry something (e.g., card, keys).

As we design our own authentication scheme, we draw on some of these lessons: we

focus on making the authentication process quick; the study reaffirms our design choice of using a wristband (study participants liked the wristband form factor and how, once worn, it stays with them); and we use the failure rates observed in the study as a benchmark when evaluating our authentication scheme.

5

Bilateral Verification

CSAW uses a different approach to user verification than existing authentication schemes; we call it *bilateral verification*. We define bilateral verification as the process of verifying the user's identity by combining knowledge from two observers who cannot verify the user's identity using their knowledge alone. In CSAW, the computer and the user's wristband are the two observers that combine their knowledge of the user's interaction with the computer during authentication.

We often use this bilateral approach to confirm events or facts in our everyday conversations. For example, consider this hypothetical conversation between two co-workers, John and Mary, in their office. John tells Mary that on his way to office in the morning, he saw someone get pulled over by a police officer near the office at Boyle Street, and he thinks it was their co-worker named Dan, but he is not sure. Mary informs him that Dan did in fact get pulled over by a police officer that morning, but she does not know where. Based on this new knowledge from Mary, John feels confident that it was Dan whom he saw pulled over at Boyle Street. Applying this analogy to CSAW, when a user interacts with a computer, the computer is aware that someone (among the multiple authorized users) is interacting with it but does not know who, and the user's wristband is aware that its wearer is interacting with a computer (among the multiple computers that the user is authorized to use) but does not know which one; combining their knowledge, the computer and the wristband verify the wristband-wearer as the computer user.

5.1 A different approach to user verification

To illustrate the differences between the user-verification approach of CSAW and other authentication schemes, let us consider an authentication scheme abstractly. A user \mathcal{U} wishes to authenticate to a computer \mathcal{C} . To authenticate, the user, claiming to be \mathcal{U} , provides an attribute \mathcal{A} to \mathcal{C} , which verifies the claim by comparing the provided attribute \mathcal{A} with a reference attribute \mathcal{R} associated with \mathcal{U} 's identity. In current authentication schemes, the computer stores \mathcal{R} (locally or on a remote server) and when the user provides \mathcal{A} , the computer has enough knowledge to verify the user; for example, in a password-based authentication scheme, \mathcal{R} is the hash of the user's chosen password, it is stored locally; during authentication, when the user provides her password (attribute \mathcal{A}), the computer

compares it with \mathcal{R} to verify the user's identity. In CSAW, the computer does not know \mathcal{R} , so given \mathcal{A} it cannot verify the user on its own; it needs the user's wristband – the second party – to provide \mathcal{R} . The computer needs the user's wristband for *every* authentication attempt; it cannot use \mathcal{R} from a current authentication attempt to verify the user in future authentication attempts, because the reference attribute \mathcal{R} is computed – from the user's wrist movement – during the authentication attempt and it is different for each authentication attempt.

CSAW is a token-based authentication scheme – the user's wristband is the token – but it differs from other token-based schemes in how it verifies the users. Some token-based authentication schemes offer only initial authentication (one-time authentication schemes), and some schemes offer continuous authentication; CSAW provides both. In one-time authentication schemes, the user presents the token (attribute \mathcal{A}) to a verifier for authentication and the computer compares it with its stored token credentials (reference attribute \mathcal{R}); for example, a user presenting her Radio-frequency Identification (RFID) card to a RFID reader for authentication. In CSAW, the computer is aware of the wristband's presence in its radio proximity, but the attribute for authentication is the user's interaction such as tapping a key on a keyboard or picking up the phone. The effort required for initial authentication in both schemes is roughly the same, but in CSAW there is no need for the user to present the token (wristband) to the computer: the wristband communicates with the computer using a medium-range radio, so the computer can recognize the wristband from a distance.

Continuous authentication is one of CSAW's goals; the purpose of a continuous authentication scheme is to determine whether the user is *using* the target computer. Wireless token-based continuous authentication schemes authenticate a user based on her distance

(proximity) to the target computer; thus, these schemes determine whether the user is *near* the target computer. Such proximity-based schemes do not work well in shared environments where there may be multiple users near the target computer or where there is a risk of leaving the computer unattended [84]. CSAW authenticates users when they are interacting with the target computer; thus, it determines whether the user is using the target computer.

5.2 Security of bilateral verification approach

A generalized bilateral verification approach for user authentication can be expressed as follows (using notation from above). While a user is interacting with a computer, two *observers*, the computer (P_1) and a second trusted observer (P_2), make independent observations of an *attribute* of the user, resulting in two independent observed attributes, \mathcal{A} and \mathcal{R} , respectively. The two observers compare their observed attributes \mathcal{A} and \mathcal{R} to verify that user; single observation (\mathcal{A} or \mathcal{R}) is insufficient to verify the user. In CSAW, the attribute is the user's interaction with the computer, and the two observations are the desktop input (\mathcal{A}) and the wrist's sensor data (\mathcal{R}). An authentication method could use a different attribute, say, the user's heart rate or bioimpedance of their wrist: the computer and the user's wristband could measure the user's heart rate or bioimpedance of their wrist using separate sensors, and they could verify the user by comparing their readings in real time.

The security of the bilateral approach depends on the choice of the attribute and how the computer compares the two observations to verify the user. The attribute should be chosen carefully such that it should be difficult for an adversary to obtain either \mathcal{A} or \mathcal{R} . The attribute should be a) difficult to measure for anyone other than the computer and the trusted second observer, and b) difficult to synthesize based on how the computer compares the two attributes for user verification. For instance, in the above example, the user's heart rate is a

not a good choice for an attribute, because the user's heart rate also can also be measured over a distance using a camera [70].

5.3 Benefits and limitations

The bilateral verification approach offers four important benefits:

1. *Resilience to physical observation and phishing*: The observed attributes (\mathcal{A} and \mathcal{R}) used in an authentication attempt are independent of any previous authentication attempt, so if an adversary does manage to steal one (or both) attributes, he cannot use them for future authentication attempts.
2. *Memorylessness*: The attributes for authentication (\mathcal{A} and \mathcal{R}) are captured automatically (by the computer and a second observer), so the user does not have to remember any secret for authentication.
3. *Effortlessness and continuity*: Because the attributes are captured automatically, the authentication scheme could be effortless for the user and could continuously verify the user.
4. *Resilience to change in user-specific attribute*: User verification is performed solely using two observations of the user's attribute taken at the same time, so any change in the user's attribute over time would not affect the authentication scheme.

There are two main limitations of bilateral approach to user verification, both related to the attribute.

1. *Attribute may not always be measurable*: If the attribute cannot always be measured when the user is using the computer, it limits when the user could be verified. For

instance, in CSAW, the attribute is the user’s inputs to the computer, so when the user does not provide any input to the computer, she cannot be verified. This limitation is similar to facial recognition user verification systems: if the user is not in the camera’s view, she cannot be verified.

2. *Attribute may require additional hardware:* The computer or the second trusted observer may require additional hardware to measure the attribute. In CSAW, the second observer (the user’s wristband) is additional hardware. This limitation, however, is not unique to CSAW. Most authentication schemes other than password-based schemes require an additional hardware: fingerprint methods require a fingerprint reader, facial recognition requires a camera (present on most laptops and smartphones, but not common on desktops), RFID schemes require an RFID token and a reader, and so forth.

The benefits and limitations of the bilateral verification approach largely depend on the choice of the attribute; with an appropriate choice, the benefits can outweigh the limitations. We believe our choice of the attribute – user’s inputs to the computer – offers strong benefits over limitations.

5.4 Related work

To the best of our knowledge, we are the first to use the bilateral approach for user authentication to computers. Others have used bilateral-like approaches to address different problems. Cornelius et al. used the accelerometer data from two sensors, while the individual(s) carrying the sensors were walking, to determine whether the two sensors were carried by the same individual [16]. Mayrhofer and Gellersen used the accelerometer data from two sensors that

were shaken together to generate encryption keys for pairing the sensors [54]. Kalamandeen et al. used the variation in Received Signal Strength (RSS) from two devices to determine whether the devices were in close proximity [39]. Narayanan et al. used the observations of the electromagnetic signals in the environment by two devices to determine the proximity of the devices [62]. In all these works, and similar work in the literature, the observations being compared are of the same type (both observations were accelerometer data, RSS, or electromagnetic signals), whereas in CSAW, the observations are of different types, which makes comparing them challenging.

6

Desktop Authentication

Desktop computers are used in large numbers at workplaces and at homes, often by multiple users. To prevent unauthorized access users authenticate themselves before using a desktop (e.g., by logging in with username and password) and deauthenticate (i.e., log out) after their use. In some workplaces, such as hospital settings, staff tell us they typically log-in close to a hundred times daily at different workstations, often for just a few moments, to update a patient record or prescribe medication. For a user who wants to access a desktop

to perform a task, authentication is an interruption in the her workflow. Authentication requires user effort: a user must exert physical effort (e.g., type her password) and/or mental effort (e.g., recall her password). Trewin et al. studied the effect of authentication on users' workflow through an in-lab user study; they found that authentication affected users' working memory and was indeed disruptive to their workflow [94]. As a result, frustrated users devise workarounds to make the authentication process more convenient for themselves, as Adams and Sasse found in their user study [1]; the workarounds include choosing simple passwords that are easy to remember and type, writing them down on a piece of paper, or choosing to disable authentication entirely, all of which leave the computer vulnerable. Rather than browbeating users to change their behavior, we need a secure authentication method that blends seamlessly into their workflow.

An authentication method should continuously verify its user, but most authentication methods, including the common authentication methods such as password-based or fingerprint-based authentication, provide only one-time authentication, i.e., only initial authentication. In one-time authentication methods, once the desktop is unlocked any access to the desktop is associated with the user who last authenticated to the desktop. If the adversary manages to defeat the one-time authentication method (e.g., by stealing user's credentials) or bypass it (e.g., by finding an already unattended logged-in desktop), he would have unfettered access to the desktop. Even in a non-adversarial setting, it is desirable to continuously verify a user to prevent accidental misuse. For instance, Koppel et al. report that physicians frequently enter data into the wrong patient's record because they thought the open record belonged to the patient they were treating; in fact, while they were away from the desktop another physician used that desktop to update a different patient's record and forgot to log out [40]. Peisert et al. state continuous authentication as the first principle of

authentication: “Identity should be verified as long and as frequently as access to a resource is permitted. If access is ongoing then identity verification should be continuous.” [68].

Deauthentication is an important aspect of authentication, but unfortunately overlooked by most authentication methods. One-time authentication methods either employ an automatic timeout-based deauthentication (after a fixed period of inactivity, the desktop automatically deauthenticates its current user) or rely on the user to deauthenticate themselves; both approaches are ineffective in the real world, as Sinclair and Smith observed [85]. Timeouts are blind to context: to a desktop, a person reading an article on the screen or a clinician talking to the patient in the midst of updating the patient’s record appear the same as a user who has walked away [87]. Users often do not log out – they either forget to log out or intentionally leave a desktop logged-in, as a workaround, to avoid logging in again [11]. We learned about the deauthentication problem from security practitioners in hospitals, where deauthentication is an important problem, but there is clearly a similar need for automatic deauthentication in other busy workspaces where desktops are shared (such as retail shops, restaurants, or airport counters). Even in workspaces where desktops are personal and not shared, users often leave their desktops logged in when they step away, leaving their desktops open to snooping and attacks by co-workers or a passerby.

Our authentication method, called Continuous Seamless Authentication using Wristbands (CSAW; pronounced ‘seesaw’), provides initial authentication, continuous authentication, and automatic deauthentication. In CSAW, initial authentication is quick and effortless – we designed it to blend easily in users’ workflow. For continuous authentication, CSAW leverages the inputs the user provides while using the desktop for her own task. CSAW supports deauthentication based on four parameters: timeout, distance between the user and the desktop, number of steps user takes when logged-in to the desktop, and when a different

user starts using the target desktop; these parameters can be used to define deauthentication policies for different contexts.

The rest of the chapter is organized as follows. Section 6.1 gives an overview of CSAW’s approach for desktop authentication; Section 6.2 and Section 6.3 present the initial authentication method and its evaluation, respectively; Section 6.4 and Section 6.5 describe the continuous authentication method and its evaluation, respectively; Section 6.6 describes deauthentication; Section 6.7 discusses issues related to deploying CSAW in real systems, potential attacks, and extension to laptops; and Section 6.8 concludes with a summary.

6.1 Overview

CSAW uses bilateral verification for both initial and continuous authentication. In bilateral verification, an *entity* provides an *attribute*, which is observed by two *observers* that compare their observations to verify the entity (see Chapter 5). In the context of user authentication to a desktop, the entity is the user; the attribute is the user’s interaction with the desktop (i.e., the keyboard and mouse inputs provided by the user to the desktop with her wristband-hand); and the two observers are the user’s wristband and the desktop.

The wristband observes the user’s interaction (event) in terms of the wrist movement, which is captured by the accelerometer and gyroscope sensors. The wrist movement data is a time series, where each sample s at time t is a tuple with six sensor readings, $s = (ax, ay, az, gx, gy, gz)$, including three-axis sensor readings from the accelerometer (ax, ay, az) and gyroscope (gx, gy, gz) sensors. Thus the wrist-movement data stream W is of the form:

$$W = (t_1, s_1), (t_2, s_2), \dots (t_n, s_n)$$

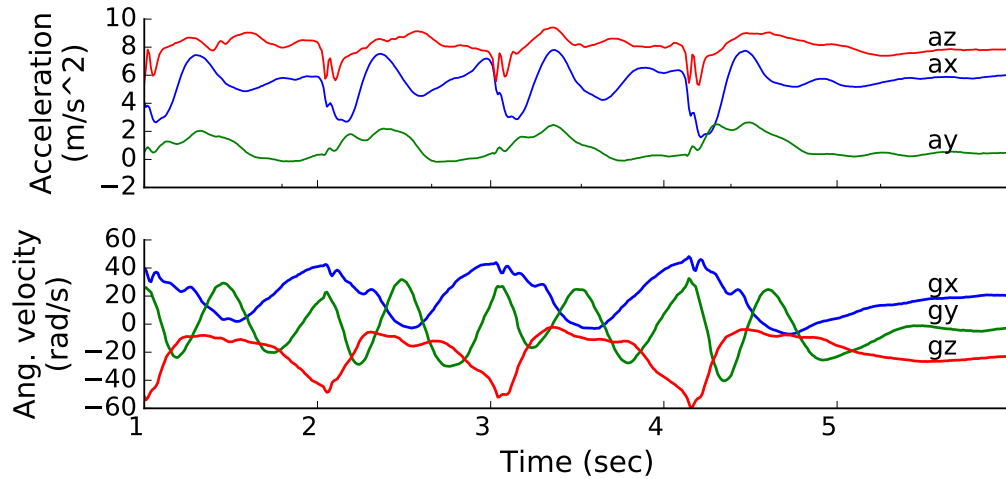


Figure 6.1: Wrist sensor data sample, from accelerometer sensor (top graph) and gyroscope sensor (bottom graph).

```

1455729785.432826, MouseMoved, 1879.078, 1087.500
1455729785.450193, MouseMoved, 1881.167, 1086.453
1455729785.752543, MouseMoved, 1895.093, 1073.222
1455729785.265091, LeftMouseDown, 1895.093, 1073.222
1455729785.449020, LeftMouseUp, 1895.093, 1073.222
1455729785.474114, KeyDown
1455729785.561697, KeyUp

```

Figure 6.2: Desktop input sample, from keyboard and mouse.

Figure 6.1 shows a six second sample of a wrist data stream, plotting acceleration and gyroscope along the six axes.

The desktop observes the user’s interaction in terms of the keyboard and mouse inputs. Figure 6.2 shows a sample log of desktop inputs; the log contains timestamp, input (corresponding to mouse movement, keystroke, mouse click), and x - and y - coordinates of the mouse pointer on the desktop display (for mouse inputs).

The desktop input data and the wrist movement data are fundamentally different semantically: one is a series of keyboard characters and mouse pointer locations on a 2D display, other is a series of instantaneous acceleration and rotation velocity values of the user’s wrist; one is a series of values that are available intermittently (when the user provides an input),

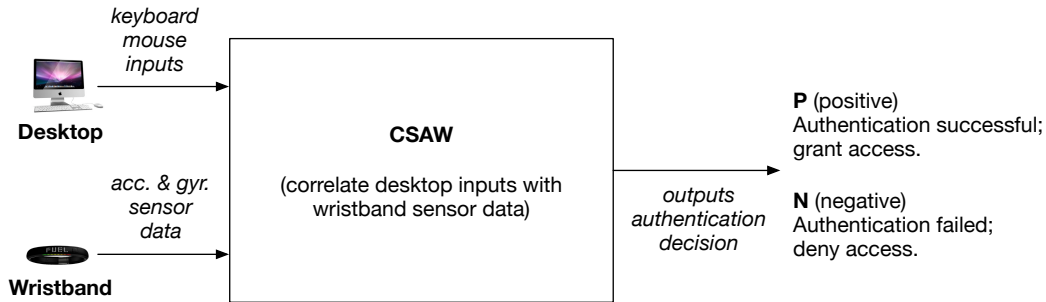


Figure 6.3: CSAW block diagram for desktops; the correlation logic is different for initial and continuous authentication.

other is a time series where values are available at a fixed rate. It is difficult to correlate these two disparate signals. We address this challenge by using the physical dynamics of a user’s wrist motion during user-desktop interaction; we identify certain types of inputs (e.g., a key tap, typing, mouse scrolling) that can be identified from the user’s wrist data, and use these events to correlate the two signals.

Figure 6.3 shows a block diagram of CSAW, generalized for initial and continuous authentication. The correlation logic for initial and continuous authentication is different, because of their different requirements for authentication: initial authentication should be quick, but continuous authentication can be slow; continuous authentication should be passive (no effort at all from the user), but during initial authentication we can ask the user to perform a simple task.

6.2 Initial authentication

One of the guiding design goals for CSAW was to integrate the authentication step into users’ workflow, while capturing their intent to access the desktop implicitly. Proximity-based authentication methods that authenticate a user when she (while carrying her authentication token) is in radio proximity are seamless and zero-effort, but they do not capture user intent

and they cannot identify the right user when multiple users are near the target desktop, which can lead to security failures in a shared-desktop environment. CSAW builds on proximity-based methods; it can recognize user intent to authenticate and identify the right user among multiple users near the target desktop.

For ease of adoption, we envision CSAW to be available as an alternate authentication method (or as a primary method, if the user chooses so) for a desktop, along with the password-based method (the backup authentication method); e.g., on PCs with fingerprint readers, Windows OS offers users the option to login with username and password or by swiping their finger. CSAW will attempt to authenticate users that chose CSAW as their primary authentication method, and if CSAW fails, the user can authenticate by entering her username and password; users who do not want to use CSAW (or who forgot their wearable device) can use their username and password for authentication.

In this section we first give the intuition for initial authentication, then describe the initial authentication protocol, and the steps involved in the protocol.

6.2.1 Intuition

There are three steps involved in accessing a typical desktop: *a*) invoke the login screen by waking up a sleeping display or disabling a screen saver, *b*) enter username and password, and *c*) start using the desktop. In CSAW, our approach is to authenticate the user based on the inputs she provides in step *a* (e.g., tapping a key), skipping step *b*. The intuition behind using key taps on a keyboard for authentication is that the user's fine wrist movement that produce the taps should strongly correlate with the keyboard inputs (i.e., taps) and based on the timing of the actions (wrist movement and taps) they should be uniquely linkable, allowing us to tie the user's wristband movement to the tap on the keyboard of a specific

desktop. Further, we can link the wristband to the wearer’s identity, using a variety of known methods, thus linking the user to the keyboard tap for the purposes of authentication. This correlation should be unique and difficult to forge (mimic by an adversary) because it involves unpredictable fine human motor actions that are difficult to observe and imitate in real time; human reaction time has been shown to average about 215 ms, which is too slow to precisely mimic a real user’s actions [76].

To illustrate, consider user U, who wears a wristband W, and hits a key on the desktop D’s keyboard; D and W are both observers of U’s keystroke, and they can communicate with each other. The desktop D receives a keystroke input from its keyboard at time t , but is unaware of the user who entered it. W determines that U entered a keystroke at time t , from the user’s wrist movement, and W provides this information to D along with credentials that identify U. If D confirms that W’s keystroke observation (timing) correlates with its own observation, it authenticates the user as U. In this simplified illustration, D and W used one *correlation point* – the timing of a keystroke – but in practice multiple correlation points are used to authenticate the user with high confidence. A correlation point is a moment during an interaction that can be detected independently by the desktop and the wristband; the desktop uses these correlation points to correlate its input with the user’s wristband movement.

For an authentication to blend easily in users’ workflow, we need an interaction that is easy and quick to perform and produces multiple correlation points; we call these *authentication interactions*. In our preliminary experiments, we explored following five potential authentication interactions:

1. *Draw-with-mouse*: The user draws a specified shape (e.g., rectangle, circle) on the display with the mouse pointer.
2. *Random-mouse-movement*: The user moves the mouse any way she likes for few

seconds.

3. *Mouse-wiggle*: The user wiggles the mouse side-to-side for few seconds.
4. *Mouse-double-click*: The user performs double click on the mouse.
5. *Tap-5x*: The user taps a key on the keyboard five times. We wanted an odd number of taps (to break ties during correlation) and wanted a tap interaction that could be easily distinguished from a double tap, which is a common interaction. Three taps is close to double tap and seven taps seemed less usable. So we chose five taps.

Among these interactions, only Tap-5x proved to be quick and easy to perform in a way that produces correlation points consistently. Draw-with-mouse was too slow to perform and it required visual attention; random-mouse-movement was easy to perform, but it proved difficult to find correlation points consistently in the random mouse movements; mouse-wiggle produced correlation points, but it was slow to perform; and mouse-double-click did not produce any correlation points (we found it difficult to identify mouse clicks in wrist movement data).

Consider, then, how user Mary may use CSAW to authenticate to her desktop. Mary purchases a CSAW-compatible wristband. Mary (M) pairs her wristband W with the target desktop D; this one-time pairing step follows any secure pairing protocol and results in D associating W with M, and enables D and W to establish mutually authentic, secure communications whenever Mary approaches (D recognizes the physical presence of W and infers the presence of Mary, M). When Mary later takes off her wristband, the wristband deactivates; when she dons her wristband she needs to enter a PIN to confirm her identity and re-activate the wristband. Later, Mary walks to the desktop and taps a key five times (Tap-5x) with her wristband hand. The desktop, after receiving the authentication interaction,

queries nearby wristbands that are paired with it, including Mary’s wristband, requesting recent wrist movement data. Mary’s wristband replies with the wrist movement data; the desktop correlates her wrist movement with the Tap-5x interaction. Mary’s wrist movement correlates and the desktop authenticates Mary. If, however, another wristband also replied to the desktop’s query and that user’s wrist movement also happened to correlate with the Tap-5x interaction, the desktop asks its user (Mary) to retry or for more information (such as her password); we discuss the details below.

6.2.2 Protocol

Recall the three use cases we described in Chapter 2 (Nancy’s use case, Jane’s use case, and the use case at the Smith memorial hospital). Jane’s home desktop is shared by multiple users – a many-to-one case. Desktops at the Memorial Hospital are shared by all users – a many-to-many case. The following CSAW authentication protocol, also shown in Figure 6.4, addresses both these use cases. We do not explicitly discuss the one-to-one case (Nancy’s use case), because it is a simpler version of the many-to-one case.

- (0) Initial step: The authentication process is initiated when a locked desktop D detects keyboard input that resembles the authentication interaction beginning at time t .
- (1) Query step: D sends a query message M_q to wristbands in radio proximity, requesting wrist movement data corresponding to the authentication interaction (from time $t - \delta$ to $t + d$, where t is the start time of the authentication interaction; δ is a small fixed duration, less than 1 s; and d is a time duration chosen to cover the authentication interaction, e.g., 5 s).
- (2) Candidate response step: Among the nearby wristbands, each wristband determines if

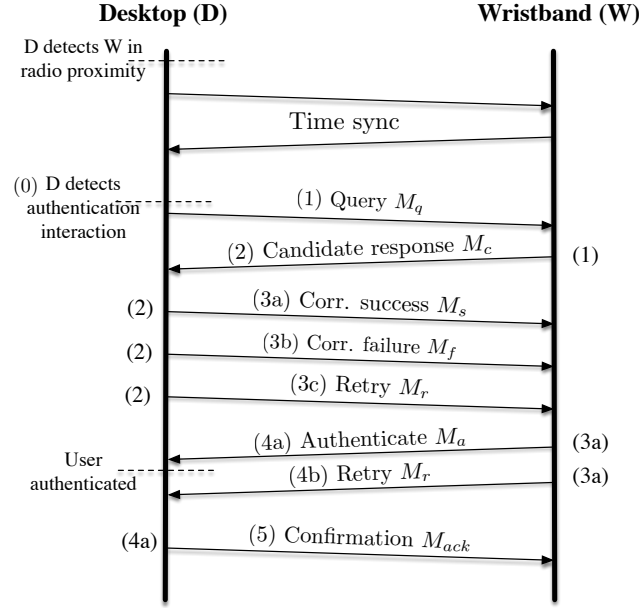


Figure 6.4: Authentication protocol. Numbers correspond to the steps in the protocol; the messages with the same number (but different letter) represent different responses the desktop/wristband can send when it receives a message, depending on some condition (e.g., when the wristband receives 3a, it sends either 4a or 4b depending on whether the user should be authenticated).

it is a *candidate* for this request; only candidate wristbands respond with the sensor data for the given period, message M_c .

(3) Correlation step: D correlates data received from candidate wristband(s) with the authentication interaction to find the best match.

(3a) If D finds only one match with high confidence, D sends a successful correlation message M_s to that wristband indicating that the user can be authenticated.

(3b) D sends a failed correlation message M_f to rest of the candidate wristbands, indicating failed authentication.

(3c) Disambiguation step: if, on the other hand, there are two (or more) wristbands that correlate with the authentication interaction, D asks its user to repeat the authentication interaction. Alternatively, D can fall back to a default authentication method like username and password. D sends a retry message M_r to both

wristbands indicating that the correlation did not succeed, but more information is required for authentication.

(4) Wristband confirmation step: In this step the wristband confirms to the desktop that the user should be authenticated. This step ensures that the user is not authenticated to multiple desktops at the same time. In a many-to-one use case (many users, one desktop), a wristband can be a candidate for only one desktop. In a many-to-many use case, a wristband might be a candidate for two (or more) desktops; such a wristband sends its motion data to all such desktops, and waits for their response.

(4a) If the candidate wristband receives a message M_s from only one desktop, it responds with an authenticate message M_a , allowing the desktop to proceed and authenticate the user.

(4b) If the candidate wristband receives M_s from multiple desktops, the candidate wristband denies authentication to all desktops with a retry message M_r , indicating that the desktops should ask the user to authenticate again, as in the step 3c.

(4c) If the candidate wristband does not receive M_s , but receives M_f or M_r , it alerts the user, indicating that an authentication attempt was made and it failed, and she should try again.

(5) Desktop confirmation step: After D receives M_a from a wristband and it has authenticated the user, D sends an acknowledgment message M_{ack} to the wristband confirming that the user has been successfully authenticated.

(6) Feedback step (optional): In CSAW, the user can configure her wristband to receive feedback on the authentication process. The wristband alerts the user on successful

authentication (when it receives M_{ack}) or if the user needs to provide more information to be authenticated (when it receives/sends M_r).

We assume the communication between the wristband and the desktop is reliable, i.e., the underlying MAC or the protocol layer in the communication stack handles message failures and guarantees message delivery. If the communication breaks between wristband and desktop, the authentication protocol aborts and the user does not get authenticated by CSAW. We discuss how this protocol helps achieve our security goals in our evaluation (Section 6.3).

6.2.3 Identifying candidate wristbands

A wristband should participate in the authentication protocol if its wearer intends to authenticate to a desktop. The user expresses her intent to authenticate by providing the authentication interaction; to quickly detect the authentication interaction, the wristband is constantly sensing its wearer's wrist movement. In CSAW, a wristband is a candidate for an authentication protocol initiated by a desktop if

1. the wristband has been paired with the desktop and is presently in radio range of that desktop,
2. the wristband owner is not interacting with another desktop, and
3. the wristband's wrist movement is similar to the authentication interaction.

The first condition excludes unauthentic wristbands and desktops, and ensures the desktop-wristband communications are conducted with integrity and confidentiality. The second condition filters out users who are interacting with another desktop and so cannot initiate authentication on the target desktop. The wristband is aware of its wearer interactions with a desktop, because the wristband and the desktop are in constant communication –

for the purpose of continuous authentication; if, however, the wristband cannot determine whether its wearer is interacting with a desktop, it assumes she is not. Users can configure their wristband to allow authentication to only one desktop at a time; so, if a user forgets to log out on desktop D_1 and later gets authenticated on desktop D_2 , the wristband would notify D_1 to deauthenticate her.

The third and the last condition filters out users that did not perform the authentication interaction, and hence could not have initiated authentication on the target desktop – for example, users who are walking near the target desktop. To filter out such users, we use a pre-trained classification model (a classifier) that determines whether a given window of wrist sensor data appears likely to be an authentication interaction. Such models can easily be implemented by a low-power microcontroller in today’s wearable technology.

In an environment where desktops are shared by many users, there may be multiple authorized users in radio range proximity of a desktop. These three criteria filter out users to form a small set of candidate users, which reduces the authentication time (by reducing the communication in the protocol) and the attack surface.

6.2.4 Correlation step

This is the main step in the authentication process. In this step the CSAW algorithm (running on the desktop) correlates the wrist movement data from candidate wristband(s) with desktop input data and determines which user, if any, should be authenticated. We correlate these two disparate signals by identifying a common event – a correlation point – that should appear in both signals, and comparing the timing of these correlation points. Thus, there are two steps involved: detecting correlation points and matching of the correlation points.

Detecting correlation points. The authentication interaction in CSAW, Tap-5x interaction, consists of five individual taps (keystrokes) on a keyboard. A tap event (or simply *tap*) begins when the key is pressed down (`KeyDown` in desktop input) and ends when the key is released (`KeyUp` in desktop input). During a tap, the direction of the user’s finger movement reverses: the user’s finger presses the key downwards (desktop receives `KeyDown` input) and as the user lifts the finger, the key recoils (desktop receives `KeyUp` input). At `KeyDown` the impact of the tap causes a vibration at the wrist, which appears as a peak in the wrist signal; at `KeyUp`, the sudden reversed movement of the finger appears as a peak in the wrist signal. Thus, the correlation points for the Tap-5x interaction are the beginning and the end of the individual taps, which appear as `KeyDown` and `KeyUp` in desktop inputs, respectively, and as peaks in the wrist signal. These correlation points are matched using the time when they appear in the desktop input and the wrist signal; thus, there are two sequences of timestamps: a sequence with the timestamps of the correlation points detected in the desktop input, S_d ; and a sequence with the timestamps of the correlation points detected in the wrist signal, S_w .

Depending on how a user performs the Tap-5x interaction and the orientation of the wristband on her wrist, she may have prominent peaks in her wrist signal at the beginning of the taps (`KeyDown`) or end of the taps (`KeyUp`). So, we correlate the peaks in the wrist signal with the `KeyDown` events and the `KeyUp` events separately, finding the best match; we get two S_d sequences, one with timestamps of `KeyDown` events and one with timestamps of `KeyUp` events. Figure 6.5 shows a Tap-5x interaction performed by a participant in our user study; the peaks in the wrist signal are prominent at the beginning of the taps.

A wristband is typically worn on the arm, just behind the wrist joint; the accelerometer sensor captures the vertical and horizontal movement (acceleration) of the wrist, and the gyroscope sensors captures any rotation at the wrist, either due to the rotation of the arm or

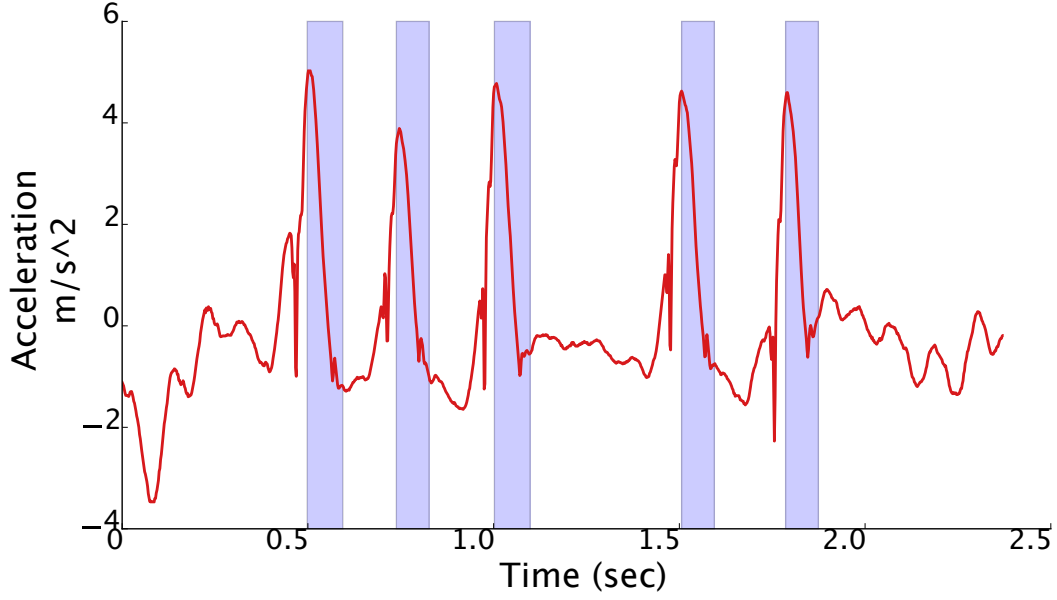


Figure 6.5: Wrist acceleration along z -axis for a Tap-5x interaction. The shaded region indicates the duration of a tap (keystroke), from `KeyDown` to `KeyUp`.

the movements of the fingers, which could get translated into minute rotations at the wrist. Depending on how a user performs the Tap-5x interaction, the peaks may be more prominent in accelerometer signal or gyroscope signal. Moreover, depending on the orientation of the wristband, the peaks in the wrist signal may appear more prominent in one axis than others. So, we find peaks in individual accelerometer and gyroscope axes, computing six S_w sequences. A peak is the highest data point in its neighborhood ($\pm\epsilon$), i.e., there is a peak at time t if its amplitude is greater than any other data point in $[t - \epsilon, t + \epsilon]$. In CSAW, ϵ is 25 ms, because the duration of a tap in the Tap-5x is about 100 ms, and we want to detect peaks occurring at the start and end of the tap (i.e., at `KeyDown` and `KeyUp`), so we look for peaks within a 50 ms neighborhood.

Matching correlation points. In this step, two sequences (S_d and S_w) are matched and a score is given to them based on how well they match. There are six S_w sequences and two S_d sequences. Each of the six S_w sequences is matched with each of the two S_d sequences,

and the best matching score is chosen as the correlation score of the wrist signal and the desktop input.

If the two sequences are of same length, then aligning and matching them is straightforward. The wrist signal is, however, very noisy and may have more or fewer correlation points than what we expect ($|S_w| \neq |S_d|$), in which case we want to match *corresponding* correlation points – timestamps that are closest to each other – in both sequences while penalizing missing or extra timestamps in the S_w sequence.

Fortunately, the problem of sequence matching is well studied in bioinformatics. We adapt the Needleman-Wunsch algorithm, which is used to align two given protein sequences [63]. Given two sequences, the algorithm produces two aligned sequences with the maximum similarity score. For example, for sequences ABCDEF and ABCGF the algorithm would output ABCDEF and ABC–GF as the two aligned sequences, for a given scoring matrix. A similarity score for the sequences ABCDEF and ABC–GF is determined by comparing letters at each position: if two letters are the same, it is a *match* and a positive match reward is added to the similarity score; if two letters are different (E and G at fifth position), it is a *mismatch* and a mis-match penalty is added to the similarity score; if there is an insertion or deletion (D and – at fourth position), it is an insertion or deletion (called *indel*) and a gap penalty is added to the similarity score. The match reward, mismatch penalty, and gap penalty are defined by a predefined scoring matrix. The Needleman-Wunsch algorithm determines the best alignment using dynamic programming.

In CSAW, we use a variation of this algorithm to align two sequences S_d and S_w ; two timestamps, t_1 and t_2 are considered a match if $|t_1 - t_2| \leq \tau$, where τ is the *matching threshold* (in CSAW, τ is 50 ms). CSAW uses a scoring matrix, where a match is 1, a mismatch is -1 , and an indel is -0.5 . Along with the aligned sequences, the algorithm

also gives the similarity score for the aligned sequences, which is computed as the sum of all matches (number of matches \times match reward), all mismatches (number of mismatches \times mis-match penalty), and all gaps (number of gaps in both sequences \times gap penalty). We use normalized similarity score as the correlation score (c); to normalize the similarity score we divide it by the maximum possible similarity score for sequence S_d with any other sequence – that is, similarity score if the actual sequence S_w is what we expected, same as S_d . A correlation score of one ($c = 1$) indicates perfect correlation between the user’s wrist movement and the keyboard inputs. If the correlation score is greater than the correlation threshold, τ_c ($c > \tau_c$), we consider the correlation good enough to authenticate the user; in CSAW, τ_c is 0.8.

6.2.5 Authentication feedback

As we develop a seamless (and near invisible) authentication, it is important to conform to the user’s mental model: the user should not be authenticated except at times and on desktops where she intends to authenticate, and she should be aware of where and when she is authenticated. In a password-based authentication method the user expresses her intent explicitly by entering her username and password, pressing the ‘return’ key, and a successful authentication is indicated when the desktop unlocks. In CSAW, the user receives similar positive feedback to a successful authentication – the desired desktop unlocks – but we go further. A CSAW wristband uses haptic feedback (vibration) to alert its wearer that her wristband is engaged in authentication at a nearby desktop.

When the wristband receives an M_{ack} message (indicating successful authentication) or sends/receives an M_r message (indicating the need to provide more information) it vibrates to alert the user. If she is authenticating to a desktop, this feedback provides confirmation

of the activity. If, however, she did *not* intend to authenticate to a desktop at the moment, this haptic feedback alerts her to the possibility that someone may be attempting to log in to a nearby workstation without her intent (e.g., when Alice is near a workstation D but not using D, and an adversary attempts to log in as Alice on D).

6.3 Evaluation of Initial authentication

In CSAW, we want initial authentication to be effortless, quick, intentional, usable, secure, and user-agnostic (our design goals, as described in Section 2.4). By design, initial authentication in CSAW is effortless: the user does not have to remember any secret (no effort to memorize or recall), and the Tap-5x authentication interaction is simple (no cognitive effort) and easy (no physical effort) to perform. To evaluate the rest of the goals we conducted two in-lab user studies: a feasibility user study, to evaluate CSAW from the system’s perspective; and the users’ perception user study, to evaluate CSAW from a user’s perspective.

6.3.1 Feasibility user study

We recruited seventeen participants for the user study: ten male and seven female; eleven graduate students, four undergraduate students, and two college employees. We explained to the participants that we were developing an authentication method and the purpose of the study is to collect data to test the feasibility of the method. We further explained how the initial authentication method works, focusing on what a user would have to do to unlock a desktop computer – wear the CSAW wristband and perform Tap-5x with the wristband hand.

We asked participants to wear a Shimmer device – the CSAW wristband for the purpose of our study – on their wrist of their dominant hand. The Shimmer is a research platform that contains accelerometer and gyroscope sensors, and a Bluetooth radio that we use to send

the sensor data to the desktop [82]; we calibrated the accelerometer and gyroscope sensor in the Shimmer prior to its use. We sampled the accelerometer and gyroscope sensors in the Shimmer at 512 Hz, and streamed the sensor data to the desktop in real time, where it was logged to a file. To capture keyboard inputs to the desktop, we wrote a Python script to intercept keyboard inputs from the desktop OS; we used Apple iMac desktops in our user study.

Wearing the Shimmer, each participant performed 40 Tap-5x interactions in four sessions, with a few minutes break between sessions; in each session the participant performed 10 Tap-5x interactions repeatedly, with a brief pause in between. To perform the Tap-5x interaction, we instructed the participants to ‘Tap a key, any key, on the keyboard five times’. We did not demonstrate or give them any specific instruction on how to perform taps; we wanted to capture participants’ natural Tap-5x interaction. We did, however, instruct them to assume that the desktop display is off and they have to wake up the display with their Tap-5x interaction; we envision CSAW being used in this way. The Tap-5x interaction can be performed while resting the wrist on the table or keeping it (floating) in the air; participants were asked to perform 20 Tap-5x interactions with each wrist configuration.

We collected two types of data for the Tap-5x interactions performed by the participants: *a)* the participant’s wrist movement, captured by the Shimmer, and *b)* the keyboard inputs received by the desktop, intercepted by the script we wrote. Clock synchronization between the desktop and the wristband is important in CSAW. In our study, we synchronize the desktop and the Shimmer clocks using the receive time of the sensor data on the desktop: we determine the offset between the clocks at the start of the Shimmer sensor data streaming, and adjust the sensor sample time to the desktop clock.

Separately, we recruited a volunteer to wear Shimmer on his wrist and collected data

for 10 min while he was sitting still on a chair (“sitting” activity); for 10 min while he was walking (“walking” activity); and for 50 min while he performed different activities (moving around, sorting things, writing, cooking, chopping vegetables, etc; “other” activity). We collected only 10 min data for sitting and walking activity, because that was enough to capture a user’s sitting or walking pattern.

6.3.2 Quickness

Authentication in CSAW is quick: in our feasibility user study participants performed the Tap-5x interaction in 1.5 s on average. The total authentication time in CSAW is the sum of the time the user takes to perform Tap-5x, the communication time in the protocol (which is negligible), the time to determine wristband candidacy (i.e., whether the wristband is a candidate), and the time to compute the correlation score. In our tests (on a laptop with 8 GB RAM and 2.6 GHz Intel core i7 processor) it took about 500 ms to determine wristband candidacy and compute correlation, which can be further reduced with optimization; in an actual implementation of CSAW the computation for determining wristband candidacy will be done on the wristband (in our proof-of-concept prototype, the wristband streamed all the wrist data to the desktop, and the desktop did all the computation). Thus, the total authentication time in CSAW was about 2 s on average.

6.3.3 Recognizing intentionality

In CSAW the user’s intent to authenticate is implicitly determined by the wristband during the candidate response step in the protocol (see Section 6.2.2); a candidate wristband represents its wearer’s intent to authenticate. If the wristband fails to detect the user’s intent to authenticate (a false negative), the user does not get authenticated; if the wristband

falsely detects the user’s intent to authenticate (a false positive), the wristband engages with the desktop in the authentication protocol, but it does not guarantee that the user will be authenticated. Thus, we want a low FNR and a low FPR, but a high FPR is acceptable, because the false-positive error can be corrected in the correlation step.

Recall that CSAW uses a pre-trained classifier to identify Tap-5x interactions (see Section 6.2.3). We use the data from the feasibility user study to train the classifier: we use wrist sensor data from participants’ Tap-5x interactions as positive samples ($n = 575$), and randomly selected 2 s wrist sensor data samples from the three activities as negative samples ($n = 60$). We intentionally added the imbalance in the training dataset (more positive samples than negative samples) to keep the FNR value low at the trade-off of a high FPR.

We evaluated CSAW’s FNR in detecting user’s intent, i.e., failure to detect a user’s intent when the user actually performed a Tap-5x interaction, by testing Tap-5x interactions (as test cases) with the trained classifier. To ensure we test CSAW’s performance in a user-agnostic manner, we use leave-one out validation: for each participant, the classifier is not trained using data from the participant that is being tested. The average FNR, across all participants, was 0.002 ± 0.003 . This means that in 1,000 authentication attempts, CSAW failed to detect the user’s intent to authenticate twice, and the user had to try again.

We evaluated CSAW’s FPR for identifying candidate wristbands, i.e., falsely detecting the user’s intent, by testing 3,000 two-second wrist signal samples selected randomly from the activity dataset (as test cases) with a classifier trained with Tap-5x interactions from all participants. This test is equivalent to asking how often, in CSAW, does a user’s wrist movement get accidentally detected as her intention to authenticate, while the user is sitting, walking, or doing other activities? Table 6.1 shows the FPR for these different activities.

Table 6.1: False-positive rate (FPR) for detecting user’s intent to authenticate when the user is sitting, walking, or doing other activities; mean \pm standard-deviation across users.

Activity	FPR
Sitting	0.000 ± 0.000
Walking	0.013 ± 0.006
Other	0.151 ± 0.021

The FPR for sitting is zero (as expected) and for walking it is low (0.001 ± 0.006). The FPR for the Other activity is high (0.151 ± 0.02), because during this activity, for some duration, the participant’s the wrist movement was similar to Tap-5x interaction – the reason why we chose this activity. As mentioned above, a high FPR rate for detecting user’s intent is acceptable, because falsely detecting intent does not imply that the user will be authenticated (as we show below).

6.3.4 Usability

In this section we evaluate CSAW’s usability from the system’s perspective, which considers an authentication method “usable” if it does not make any error in authenticating the right user, i.e., it has zero FNR. In authentication methods such as passwords, fingerprint, or face recognition, authentication is performed solely based on the credentials the individual presents to the desktop; it does not depend on the activities of other users near the desktop. In CSAW, however, the presence of other users near the target desktops does make a difference for a user’s authentication to the target desktop. So, to evaluate CSAW, we consider two scenarios, based on what other users are doing when the user (Alice) performs authentication: *single-login scenario* and *multiple-login scenario*.

Single-login scenario. In this scenario, only one user (Alice) performs authentication; there is no adversary. We consider two versions of this scenario, based on how many users

are present near the target desktop: *single-user case*, there is no other user near the target desktop, except Alice; and *multi-user case*, there are other users near the target desktop, but they do not perform authentication. We want low FNR for both the single-user case and the multi-user case.

To evaluate CSAW’s FNR for single-user case, we fed wrist data and keyboard data for Tap-5x interactions from all participants to CSAW, as authentication attempts (test cases); based on CSAW’s output, we mark each test case as false-negative (failed authentication) or true-positive (successful authentication). In the multi-user case, when Alice attempts to authenticate, the desktop sends out a query to all nearby users, and it gets responses (wrist data samples) from nearby users. The desktop correlates each wrist sample with its keyboard-input sample and authenticates by comparing the correlation scores. We tested the multi-user case with each participant in the role of Alice and we used 2 s wrist data samples from the activity dataset as Bob’s wrist data; we fed the two wrist data samples and one keyboard sample to CSAW, and based on its output, we mark the test case true-positive (Alice gets authenticated) or false-negative.

Figure 6.6 shows the false-negative rate for each participant in user study for the single-user case and the multi-user case. The mean FNR for the single-user case, across all participants, is $0.025 (\pm 0.061)$; this FNR includes the error in recognizing user’s intent (from Section 6.3.3). In our feasibility user study, we did not instruct participants on how to perform the Tap-5x interaction (e.g., how to move the wrist or how fast or forceful the tap should be); we simply told them to tap a key five times keeping the wrist in the air, because we wanted to evaluate how CSAW performs with the natural tap interaction when the wrist is in the air. CSAW worked perfectly (zero FNR) for 14 of 17 participants; for three participants, CSAW failed a few times; they performed the Tap-5x interaction gently, with

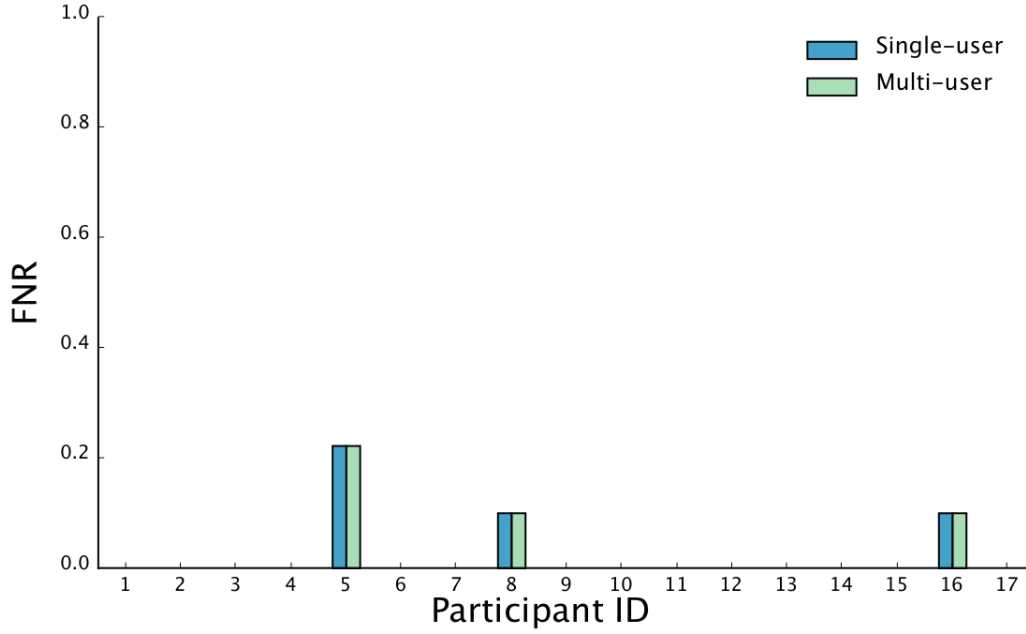


Figure 6.6: False-negative rate (FNR) for each participant, when there is one user (multi-user case) and no other user (single-user case) near the target desktop.

an immobile wrist, and the peaks corresponding to the `KeyDown` and `KeyUp` movement were not prominent enough in the wrist signal for CSAW to identify and correlate. With some training, we expect CSAW to perform better for most users.

The FNR for the multi-user case was the same as the single-user case, $0.025 (\pm 0.061)$, which suggests that having a second user (who is not using a desktop or performing a Tap-5x-like action) near the target desktop did not affect CSAW’s accuracy: CSAW was able to correctly identify the user among two users. Thus, in a busy multi-user environment, a user can simply walk to a desktop and authenticate with the Tap-5x interaction, without having to specify the username.

Multiple-login scenario. In this scenario, two or more users attempt to log in at distinct (but nearby) desktops; this scenario is similar to a multi-user shared desktop environment in hospitals, where clinicians are frequently logging in to desktops, maybe around the same

time. We assume there is no adversary in this scenario; we consider an adversary who uses simultaneous login for attack in the next section.

For two users, this scenario can be expressed as follows. Alice attempts to authenticate to desktop D_1 at time t_1 , and Bob attempts to authenticate to desktop D_2 at time t_2 . When Alice attempts to authenticate and if Bob has not authenticated by then ($t_2 > t_1$), desktop D_1 will query for wrist data from Alice's and Bob's wristbands, and their wristbands respond with the wrist data, and D_1 will correlate the authentication interaction input with their wrist movements. As per the authentication protocol, if the correlation succeeds with both the wrist bands, D_1 will ask Alice to retry; if the correlation succeeds only with Alice's wristband, she will be authenticated; if the correlation succeeds with only Bob's wristband, Bob will be authenticated instead; and if the correlation fails with both the wristbands, Alice will be asked to retry. The ideal outcome in this scenario is for D_1 to authenticate Alice (true positive); the worst outcome is for D_1 to authenticate to Bob (false positive); the outcomes where Alice is asked to retry (false negative) are acceptable if they does not happen frequently.

In CSAW, the outcome depends on Bob's wrist movement at t_1 . If Bob logs in at the same time as Alice ($dt = |t_2 - t_1| = 0$), then Bob's wrist movement would be similar to Alice's wrist movement, since both will be performing the Tap-5x interaction. To evaluate this particular case ($dt = 0$), we extracted data from two Tap-5x interactions, from two participants in the feasibility user study (one as Alice and one as Bob), and we adjusted the start time of Bob's interaction to match the start time of Alice's interaction ($dt = 0$). We repeated this experiment with each participant in the role of Alice and every other participant in the role of Bob, for every pair of Tap-5x interactions. For this particular case, we want a low FPR, and preferably low FNR, but a high FNR is acceptable, because such an event

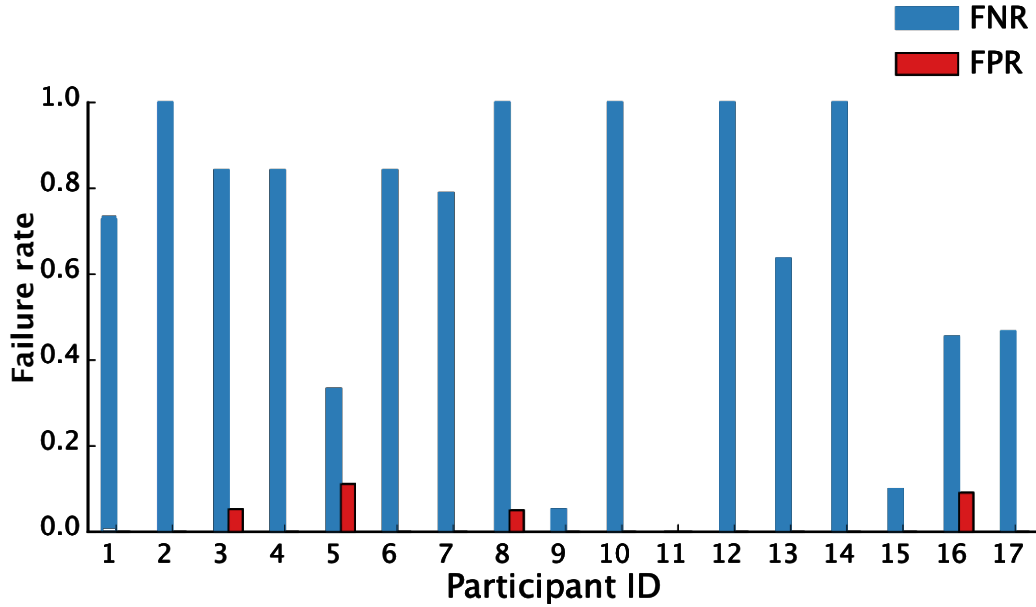


Figure 6.7: False-negative rate (FNR) and false-positive rate (FPR) when two users attempt to authenticate at the exact same time on nearby desktops.

($dt = 0$) would not happen frequently in practice – in real settings we expect at least a few seconds difference between two user’s login attempts.

Figure 6.7 shows the FNR and FPR for each participant as Alice. The FNR rate is high for most participants, which indicates that these participants perform Tap-5x interactions the same way, and hence, Bob’s wrist movement correlates with Alice’s Tap-5x interaction. CSAW will ask Alice to retry and next time, it is less likely that Bob will also perform Tap-5x interaction at the same time as her second attempt. For such an unlikely case, it is acceptable to have a high FNR. It is important to have a low FPR – we do not want Bob to get logged in on Alice’s desktop, and CSAW does achieves a low FPR of 0.018 (0.036), mean (standard deviation) across all participants.

6.3.5 Security

We are concerned about impersonation attacks, as we described in our adversary model (see Section 2.3). We focus on two types of impersonation attacks:

1. *Accidental Impersonation attack*: The adversary waits for opportunities when the victim is in the radio proximity of the target desktop, and attempts to authenticate to the target desktop as the victim user, by providing Tap-5x interactions arbitrarily, hoping that the interactions would correlate with the victim's wrist movement.
2. *Malicious Impersonation attack*: The adversary visually monitors the victim user; when the victim is attempting to authenticate to a nearby desktop, the adversary tries to mimic her wrist movement and performs the Tap-5x authenticate interaction on the target desktop, at the same time she performs it on the nearby desktop. The adversary may also anticipate when her wrist will move in a pattern similar to Tap-5x (while she is not authenticating to a desktop), and during that moment perform Tap-5x on the target desktop.

We consider two scenarios, an *accidental-login scenario* and *malicious adversary scenario*, that represent these two attacks.

Accidental-login scenario. In this scenario, Alice (the user) is near the target desktop, but she is not authenticating to any desktop; Eve (the adversary) attempts to log in the target desktop, impersonating Alice. We do not want Alice to get accidentally logged in to the target desktop, just because she was near the target desktop (as would happen in all existing proximity-based methods); we want FPR close to zero for this scenario. Eve's success depends on Alice's wrist movement during his authentication attempt and how well Eve matched his authentication interaction with Alice's wrist movement.

We evaluate CSAW for this scenario using the data from our feasibility user study: we consider each participant in the role of Eve, who provides the Tap-5x interaction, and we consider the participant who provided the activity data (sitting, walking, other) in the role of Alice. We test each authentication interaction from Eve with 1,000 random samples of Alice’s wrist movement data from each of the three activities; we adjust the timestamp of Alice’s wrist data to match Eve’s authentication interaction.

The FPR for all our tests was zero; CSAW prevented all accidental login scenarios we tested against (we thus do not include a plot). A majority (about 85%) test cases get rejected because Alice’s wrist data did not indicate her intent to authenticate, and the remaining tested cases failed to correlate with the timing of the taps in Eve’s Tap-5x interaction. Thus, for the activities we test CSAW prevented all accidental authentications.

Malicious adversary scenario. In this scenario, Eve waits for Alice to attempt to log in to a nearby desktop, say D_1 , and he attempts to log in, as Alice, to the target desktop D_2 , by mimicking Alice’s authentication interaction.

We are our own best mimickers – a user is more likely to repeat a Tap-5x interaction that is more similar to her last Tap-5x interaction than a skilled adversary trying to mimic the user’s Tap-5x interaction in real time. To evaluate CSAW for this scenario, we fed CSAW a negative test case with a Tap-5x interaction desktop input from a participant and wrist signal from a different instance of a Tap-5x interaction by the same participant, adjusting timestamps so $dt = 0$. Figure 6.8 shows the FNR and FPR for each participant. The FNR is high, but this is a rare case, so a high FNR is acceptable; indeed, it is an attack scenario. It is vital to have a low FPR for this attack scenario, and CSAW achieves a low FPR (0.023 ± 0.063).

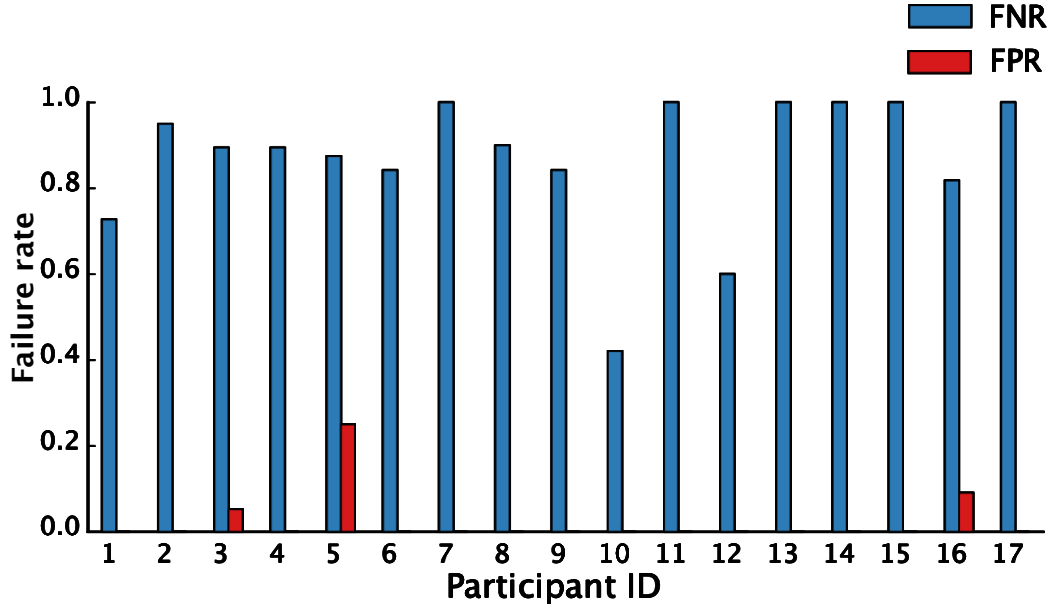


Figure 6.8: False-negative rate (FNR) and false-positive rate (FPR) for self mimicking.

The difference between the simultaneous-login scenario and this scenario is that Bob (from the simultaneous login scenario) is not a malicious adversary, trying to mimicking Alice’s Tap-5x interaction. We can see this difference because the simultaneous-login scenario has a lower FNR (0.652 ± 0.354) compared to this scenario (0.869 ± 0.160) – in CSAW the better the mimicker the higher the FNR, because CSAW denies authentication if both the user and the adversary produce wrist movement that correlates with the Tap-5x interaction.

6.3.6 Users’ perception of CSAW

So far we presented CSAW’s performance from the system’s perspective. Adoption of an authentication method, like any other security system, depends on whether its users perceive it as usable and secure. In this section we present methodology and findings of our an in-lab user study that we conducted to understand users’ perception of CSAW’s usability

and security.

Study methodology

We recruited eight participants for this study: five female and three male; four undergraduate students and four graduate students. Participants were asked to fill out a survey about their demographic information, their current preferred authentication method, and whether they prefer to tap a key or use the mouse to wake up a display (or disable a screen-saver) on a desktop/laptop they want to use. After the initial survey, participants performed 30 authentication attempts, with three different authentication methods (discussed below). We then conducted semi-structured interviews with the participants to gain insight into what they liked and disliked about each method (provided in Appendix B.1). We took detailed notes, summarizing participant responses and writing direct quotes. Finally, participants completed a modified System Usability Scale (SUS) survey for each of the three authentications methods (provided in Appendix B.2). We modified the SUS survey [14] by changing ‘system’ to ‘method’, and removing the question ‘I found the various functions in this system were well integrated’, because it was not applicable to the authentication methods in the study.

Participants are known to be biased in user studies and they try to give answers they expect the researcher is looking for: We have experienced this bias in our previous user studies – some participants ask, “Is this what you are looking for?” [9]. To reduce this bias we chose participants who were not aware of this work, had not participated in the feasibility study, and we did not inform the participants that were evaluating our developed authentication method. We told participants that we were evaluating two potential authentication methods that other researchers have proposed: a) mouse-based method (a fictitious method based on mouse movement dynamics, i.e., a behavioral biometric method), and b) tap-based method

(CSAW). We explained how both methods work. In the mouse-based method two objects are shown on the screen and the user drags one object over another, and the user is authenticated based on how she moves the mouse – her unique mouse movement signature; and in the tap-based method the user wears a wristband (which acts as the user’s identity) and taps a key five times, and the user is authenticated based on the timings of the taps that the wristband determines from the wrist motion, which it sends it to the desktop. We told participants that we are collecting data to study the feasibility of these methods and to get their feedback on their perception of the usability and security of these two methods. In addition to the two authentications methods, we also asked participants to perform authentication attempts with a username and password, as a baseline for comparison.

We simulated the authentication environment through a browser: participants were shown a webpage asking them to do a task required by the authentication method and then click the login button to authenticate; the webpage had written instructions for the task. For username and password method, the task was to enter a username and password (chosen at the start of the experiment); for the mouse-based method, the task was to drag one rectangle (specified on the webpage and always the same rectangle) over the second rectangle (both rectangles always appeared in same location on the webpage); and for the tap-based method, the task was to tap a key five times (any key; most users chose the space bar). During the experiment, participants performed 30 attempts, 10 for each method, in a randomized sequence (the sequence was same for each participant).

Findings

The SUS scores for the passwords, the fictitious mouse-based authentication method, and the tap-based method (a.k.a. CSAW) are shown in Figure 6.9. Since we used nine questions

in our SUS survey, we report the usability scores out of a total of 90. A higher SUS score indicates that a system is more usable. Five out of nine participants rated CSAW more usable than passwords; the average SUS score for CSAW was about the same as passwords (65.83 ± 16.34 vs. 67.5 ± 12.3). Six participants said they liked the CSAW the most among the three methods, and comment on how quick and easy it was to perform the taps.

“It’s very non-intrusive. I don’t have to think about authentication.” (P4)

Two participants rated CSAW less usable (mean score of 47.5); when inquired, one participant said he was just more comfortable and familiar with typing passwords; and the second participant did not like that he would have to wear a wristband. Several participants did not have a smartwatch or wristband; when queried about why they do not wear one, almost everyone said that they do not mind wearing one, they just have not found the one they like or could afford; they expressed willingness to wear a smartwatch if it had an utility like authentication.

Although several participants confessed that performing the mouse drag movement was simpler, they rated mouse lower than passwords (and lower than CSAW) in terms of usability, because they were more comfortable and quick with typing (and Tap-5x) than using a mouse (58.8 ± 8.75 vs. 67.5 ± 12.31).

“Using the mouse felt like a task to me.” (P2)

We also asked participants to rate how secure they thought CSAW and mouse-based method would be compared to their current based method, which was password for everyone. Five out of eight participants felt CSAW was almost as secure or more than passwords. One participant explained that she felt CSAW was more secure because of the physical device (wristband) that stays with her.

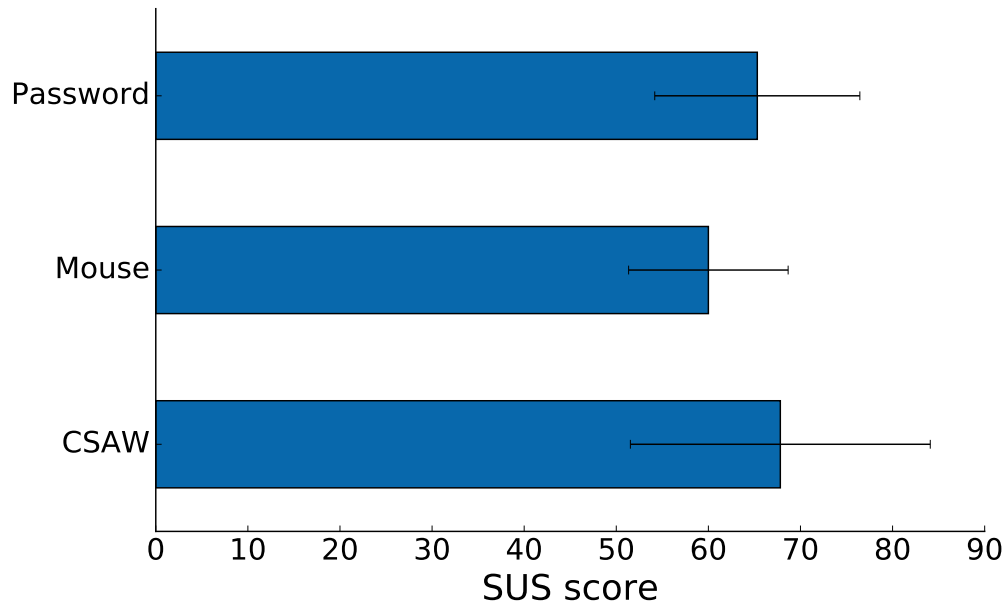


Figure 6.9: Average system usability scale (SUS) score for password-based authentication, mouse-based authentication, and CSAW; error bars show standard deviation

“Having the wristband makes it feel more secure” (P1)

Three participants rated CSAW as less secure than passwords; when we inquired, they expressed lack of confidence in the method.

In summary, all participants, except two, liked CSAW for its quickness and simplicity; some participants expressed concerns about having to wear something. We expect people who already wear a fitness band or smartwatch would be willing to use CSAW. It is difficult to conclude, from our participants’ responses, whether they perceive CSAW as being more secure than passwords. Some participants had concerns about its security, but it was due to their lack of confidence in the method. Perhaps after knowing more about how CSAW works, these participants might have expressed more confidence, but we intentionally did not share more information about CSAW compared to the fictitious mouse-based method during our study (we wanted to avoid bias for CSAW, so we gave same level of information about both methods). We expect, after using CSAW and becoming familiar with how it works,

users would feel confident about its security.

6.4 Continuous authentication

After a successful initial authentication, it is important to continuously verify the user while she uses the desktop. Continuous authentication should be passive, i.e., it should be not require any explicit input from the user, otherwise the user's workflow will be interrupted. In this section we first give an intuition for CSAW's approach for continuous authentication, and then describe the method.

6.4.1 Intuition

Unlike our approach in initial authentication, where we asked the user to perform a specific interaction (Tap-5x), we cannot ask the user to perform any action for continuous authentication. For continuous authentication, we leverage the inputs the user provides while using the desktop for her own task. These desktop inputs, however, are arbitrary as they depend on the task the user is performing on the desktop, which makes it challenging to correlate them with the user's wrist movement; for initial authentication, we used a known input for correlation – the Tap-5x interaction.

We make two observations about a user's interaction with a desktop: *a)* user-desktop interaction is often interlaced with keyboard and mouse use (especially in the clinical settings), and *b)* a wristband on the user's hand operating the mouse can be used to identify (and correlate) the type of interaction (keyboard or mouse) the user is performing. For example, when the user is scrolling or clicking, her fingers move but her wrist is relatively stationary; when the user clicks the mouse and then starts typing on the keyboard (typically with both hands), her hand will move from the mouse to keyboard. We hypothesize that a

user's wrist movement is different for different type of inputs (keyboard or mouse input); this hypothesis drives CSAW's continuous authentication.

Figure 6.10 shows a user's wrist acceleration when she was interacting with the desktop. The x -axis represents the time (in seconds) from the start of the experiment and the y -axis represents the magnitude of the acceleration, as measured by the wristband on her wrist. We marked, with shaded regions, three types of user interactions in the figure: *scrolling*, *typing*, and *MKKM*, where MKKM stands for 'Mouse to Keyboard or Keyboard to Mouse' – an interaction representing the action of switching between keyboard and mouse. As shown in the figure, the user scrolled the mouse at 65.5 s, from 66 s to 74 s, and then briefly around 75 s. The graph shows that her wrist was relatively still during scrolling, as one would expect. When she moved her hand from mouse to keyboard (around 77 s) to type, we see a sudden spike in acceleration caused as she lifted her hand off the mouse and as she rested her hands on the keyboard. As she typed (77.5 s to 83.4 s), we see small changes in the acceleration, implying that her wrist moves little during typing. After typing she switched from keyboard to mouse (around 83.5 s), and we see another sudden spike in the acceleration. The acceleration data that is not marked in the graph represents the user's other interactions with the desktop such as mouse-dragging, clicking, or hand movements not involving interaction with the desktop; we highlighted only three types of interactions on the graph for readability.

We can see the differences in the acceleration patterns between interactions. For instance, broadly speaking, there is more wrist movement during typing than scrolling, but less than when she switches between keyboard and mouse. This example supports our hypothesis that we can generate a sequence of interactions from a user's wrist movement.

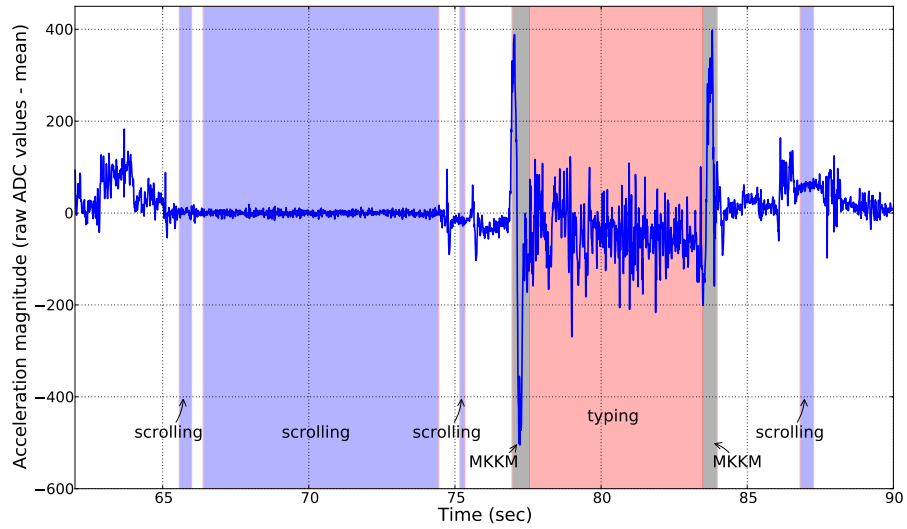


Figure 6.10: Acceleration of user's wrist when she is using a computer desktop.

6.4.2 Architecture

There are five main components in CSAW, as shown in Figure 6.11. The *interaction extractor* extracts interactions from a user's keyboard and mouse inputs, and sends the sequence of interactions to the correlator and the time intervals of the interactions to the segmenter. The *segmenter* segments the accelerometer and gyroscope data into blocks based on the time intervals it receives from the interaction extractor. The *feature extractor* extracts features for each block of data that it receives from the segmenter. The *interaction classifier* takes these features and classifies them into one of our specified interactions. The *correlator* compares the actual sequence of interactions that it receives from the interaction extractor and the inferred sequence of interactions that it receives from the interaction classifier, and it makes a decision whether the two users – the current desktop user and the user wearing the wristband – are the same (positive output) or different (negative output).

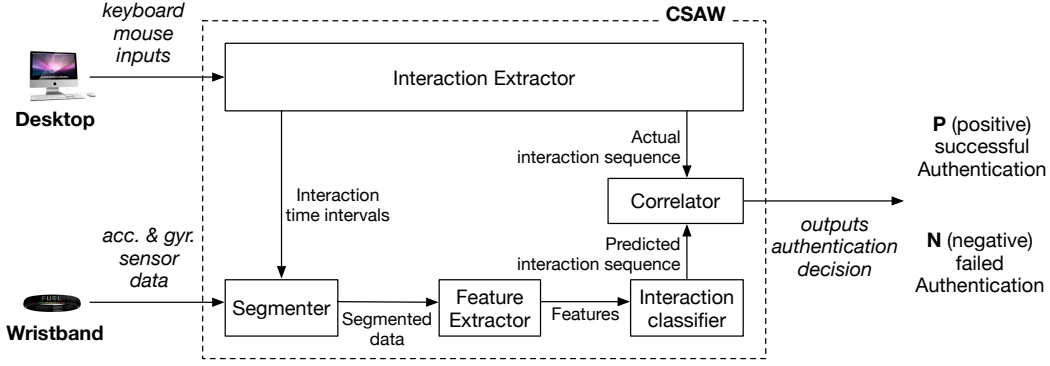


Figure 6.11: CSAW architecture for continuous authentication.

6.4.3 Interaction extractor

As mentioned, this component extracts ‘interactions’ from the input events stream generated by the OS when the user provides inputs to the desktop via keyboard or mouse. We use three main types of user interactions with a desktop for continuous authentication: *MKKM*, *scrolling*, and *typing*. There are other interactions, such as moving the mouse, dragging the mouse, or clicking the mouse, but we do not consider them because in our preliminary evaluation they did not contribute to CSAW’s performance.

MKKM. This interaction captures the users’ dominant hand (here, we mean the mouse hand) movement when she switches from the mouse to the keyboard or from the keyboard to the mouse; *MKKM* is short for ‘Mouse to Keyboard or Keyboard to Mouse’. An *MKKM* interaction consists of a mouse-related event followed by a keypress event or vice-versa.

There is, however, a challenge in identifying whether the keypress event followed by a mouse-related event was caused by the dominant hand or the other hand, because the user may press a key with her non-dominant hand while keeping her dominant hand on the mouse. With one wristband, we cannot identify such events with certainty. We account for such events by dividing the keys on the keyboard into three zones, depending on which hand the user is likely to use to press that key: left zone, middle (or ambiguous) zone, and right

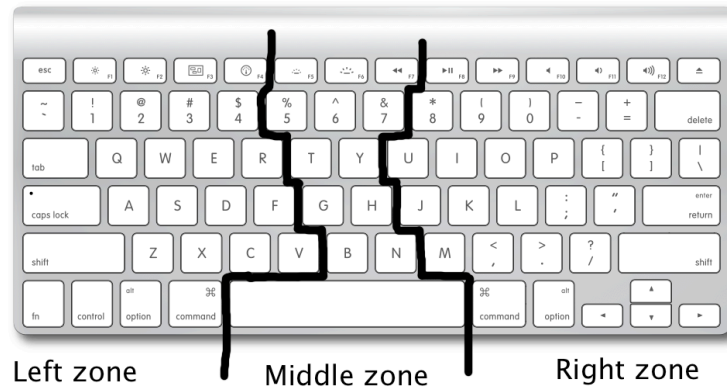


Figure 6.12: Keyboard divided into left, middle, and right zones.

zone, as shown in Figure 6.12. We introduced the ‘middle’ zone because not everyone types according to two-handed typing guidelines, which divides the keyboard into two zones, and we noticed some participants used either hand to type the keys in the middle zone. So, if the user is right handed (that is, uses the mouse with her right hand) and presses a key in the right zone after a mouse event, we assume she moved her dominant hand. Thus, we assume that users will stick with our zone divisions, i.e., use their left hand for keys in the left zone and their right hand for keys in the right zone. Some users may break this assumption, but this heuristic seemed to work well, because to identify MKKM we only need the user to press any one key in the right (or left) zone with their right (or left) hand, and we observed that all our participants did use two hands when typing.

Scrolling. This interaction captures users’ use of a scroll-wheel built-in to the mouse. When a user is scrolling, `ScrollWheel` events are continuously generated by the OS, each event reporting the amount of scroll performed by the user since the last scroll, so that the application can update the UI accordingly. We define a *scrolling* interaction as a sequence of

uninterrupted `ScrollWheel` events.

However, sometimes the mouse is slightly moved during scrolling (intentionally or accidentally), and we observe some `MouseMove` events in the `ScrollWheel` events stream. The idea behind this interaction is to capture the durations during which the user was using the mouse and her hand (wrist) was relatively stationary, so we ignore small mouse movements. We consider a mouse movement as small if the associated `MouseMove` events in the `ScrollWheel` events stream are few in number (e.g., 5 events) and the cumulative mouse displacement indicated by these `MouseMove` events is small (e.g., 5 pixels). These thresholds (minimum number of `MouseMove` events and maximum mouse displacement) are parameters in our experiments.

Thus, we define a *scrolling* interaction as a sequence of `ScrollWheel` events with few intervening `MouseMove` events such that the total mouse displacement is small (below a certain threshold).

Typing. This interaction captures the users' use of the keyboard. When a user hits a key, she first presses the key down, and then as she removes her finger she releases the key up. Associated to these actions, two events are generated by the OS for each keypress: `KeyDown` and `KeyUp`. Thus, we define a *typing* interaction as a sequence of `KeyDown` and `KeyUp` events.

If there is a continuous keypress events stream with mouse-related events in between, we count those keypress events as separate typing interactions, separated by the mouse-related events. Unlike scrolling, where we ignored small numbers of mouse-related events, during typing any mouse-related event implies that the user moved her hand from the keyboard to the mouse, which is an MKKM interaction. Thus, for a keypress events sequence with

few mouse-related events in between, we extract at least four interactions: typing, MKKM (to switch to mouse), MKKM (to switch back to keyboard), typing, and maybe scrolling between the two MKKM events if the user scrolled the mouse-wheel.

Extraction

When extracting interactions from input events, we apply three constraints: *idle threshold*, *minimum duration*, and *maximum duration*. Idle threshold is the maximum time difference between two consecutive events in an interaction. The rationale behind this constraint is to capture only the interactions that involve the user's continuous interaction with the desktop and eliminate interactions during which the user does tasks other than using the mouse and the keyboard. During a pause, there is no input to the desktop; we do not know what the user is doing, and thus cannot correlate with the user's wrist movement. If there is a pause greater than the threshold, we split the interaction into two interactions separated by the pause. For example, if in a series of keypress events there is a 2 min pause, then we split these keypress events into two *typing* interactions, one before the pause begins and one after the pause ends.

The other constraints refer to the minimum and maximum duration of interactions. If an interaction lasts for less than the minimum duration, we ignore it, and if an interaction exceeds the maximum duration we split it into two consecutive interactions. While splitting the interaction we do ensure that the new interaction is longer than the minimum duration: if the new split interaction has duration less than the minimum duration, we do not split the interaction; thus, we can have interactions that are almost as long as *minimum duration* + *maximum duration*.

Based on these three constraints and the definitions of the interactions described above, this component outputs a sequence of interactions from given input events. This sequence of

interactions, IE , is of the form

$$(I_0, t_0, t_1), (I_1, t_2, t_3), \dots$$

where I_0 is an interaction identifier (corresponding to one of the three described interactions) that starts at time t_0 and ends at t_1 , and similarly interaction I_1 spans (t_2, t_3) .

From the sequence IE , the interaction identifier sequence (I_0, I_1, \dots) is sent to the correlator. The interaction timings sequence $((t_0, t_1), (t_2, t_3), \dots)$ is sent to the segmenter.

6.4.4 Segmenter

This component receives accelerometer and gyroscope data from the user's wristband. The accelerometer data is of the form

$$(t_i, x_i, y_i, z_i), (t_j, x_j, y_j, z_j), \dots$$

where (t_i, x_i, y_i, z_i) represents one acceleration data sample taken at time t_i and the instantaneous accelerations along x , y , and z axes are x_i, y_i, z_i , respectively. The gyroscope data is of the similar form

$$(t_i, a_i, b_i, c_i), (t_j, a_j, b_j, c_j), \dots$$

where (t_i, a_i, b_i, c_i) represents one gyroscope data sample taken at time t_i and represents the instantaneous rotational velocity along x , y , and z axes, a_i, b_i, c_i , respectively.

The segmenter receives actual interaction time intervals from the interaction extractor. The segmenter breaks the acceleration data stream into blocks corresponding to each time interval, using the time of each data sample and the time of the intervals. For the time-

interval sequence, $((t_0, t_1), (t_2, t_3), \dots)$ this component will place all the accelerometer and gyroscope data samples with time $t_0 \leq t \leq t_1$ into the first block, all the data samples with time $t_2 \leq t \leq t_3$ into the second block, and so on. These data blocks are sent to the feature extractor. Acceleration and gyroscope samples outside interaction intervals are discarded.

In most signal-processing algorithms, data is segmented into blocks (also called *windows*) of equal size, but in our case the block sizes are variable. There are two main reasons to perform segmentation this way. First, when the user is not interacting with the desktop, we do not have any interaction sequences to use for authentication, so we ignore the sensor data for durations when she is not interacting with the desktop. Second, the user's interactions themselves are of variable duration so it makes sense to chunk accelerometer data this way. For the durations when she is interacting, one could segment sensor data into blocks of equal size and infer an interaction for each block, but given that a user's interactions are of variable duration, it is likely that one sensor data block would contain data for one or more interactions, which would reduce the classifier performance. Variable segmentation ensures that each sensor data block contains data for just one interaction.

6.4.5 Features

This component receives sensor data in blocks, and it computes a feature vector over each block. We do not know the orientation of and just use the magnitude of acceleration and angular velocity. For each acceleration data sample (t, x, y, z) , the magnitude m is given by

$$m = \sqrt{x^2 + y^2 + z^2}$$

and for each gyroscope data sample (t, a, b, c) , the magnitude r is given by

$$r = \sqrt{a^2 + b^2 + c^2}.$$

After computing these magnitudes, we now have for each block a series of magnitudes $(m_0, r_0), (m_1, r_1), \dots$

We compute the following 12 features over each series of acceleration and angular velocity magnitudes in a segmented interaction block: *mean*, *median*, *variance*, *standard deviation*, *median absolute deviation (MAD)*, *inter-quartile range (IQR)*, *power*, *energy*, *peak-to-peak amplitude*, *auto-correlation*, *kurtosis*, and *skew*. We chose the first seven features because others have used them successfully for activity recognition [75] and for correlation among different accelerometer signals [16]. We add the latter five features to capture the patterns of the three interactions that we noticed. During MKKM, there is a sudden spike in positive and sometimes in negative direction, so we use *peak-to-peak amplitude*. Because the placement of the peaks in a MKKM is towards the start of the interactions, we use *skew* as a feature. During typing, the peakedness is distinct, and so we included *kurtosis* as one of our features. Skewness and kurtosis are a measure of the data's variability; skewness is a measure of symmetry, or more precisely, the lack of symmetry, and kurtosis is a measure of whether the data are heavy-tailed or light-tailed relative to a normal distribution [64]. The wrist movement pattern during a typing or scrolling interaction should be roughly similar, unlike MKKM, so we use the *auto-correlation* feature to capture that difference.

For each block of data, we compute a feature vector $F = (f_0, \dots, f_{11})$, and send the sequence of feature vectors F_0, F_1, \dots (each corresponding to one interaction block) to the

interaction classifier.

6.4.6 Interaction classifier

The classifier takes a feature vector F as input and outputs an interaction identifier, its inference that the sensor data associated with that feature vector represents that interaction.

To train the classifier, we use data from our in-lab user study; we segment a participant's wrist sensor data based on her actual interaction timings, as described above. We feed the classifier with feature vectors corresponding to the actual interaction and provide the actual interaction labels. Later, when evaluating our approach with a given participant, we use a classifier that was trained with other participants' data, because our intent is for the classifier to be user agnostic.

The classifier receives a sequence of feature vectors and it outputs its inference, a sequence of interaction identifiers (i_0, i_1, \dots) . It then sends this sequence to the correlator.

We explored two classifiers: Naive Bayes classifier and Random Forest classifier. For our dataset, the Random Forest classifier outperformed the Naive Bayes classifier; the results reported in Section 6.5 are with the Random Forest classifier.

6.4.7 Correlator

The correlator matches two sequences: the sequence of actual interactions and the sequence of interactions inferred by the classifier based on the user's wrist movement. If the two sequences match, the correlator outputs 1 indicating a successful authentication, i.e., the current desktop and the wristband user are the same. On the other hand, if the two sequences do not match, it outputs 0 indicating a failed authentication, i.e., the current desktop and the wristband user are different.

To match the two sequences, we use four parameters: window size, overlap fraction, match threshold, and grace period. The window size, w , is the number of interactions the correlator compares at a time. The overlap fraction, f ($0 \leq f < 1$), indicates how much we should overlap the moving window, 0 being no overlap. For each window the correlator computes a matching score (between 0 and 1) indicating how well the two sequences match in that window; 0 being no match at all, and 1 being a complete match. If the matching score for a window is greater than the match threshold, m , we output 1 for that window, indicating a successful authentication for that window. Otherwise, we output 0 for that window.

If we incorrectly output 0 for a window and deauthenticate the user immediately, it would frustrate the user. To account for such false negatives, we introduce the grace period parameter, g . This parameter indicates how many consecutive window scores of 0 are measured to deauthenticate the user. For example, if $g = 3$ then we should get 0 for three consecutive windows before we deauthenticate the user. We reset the zero-count when we get a window with output 1. This parameter increases convenience but also increases security risk; we keep its default value low.

6.5 Evaluation of Continuous Authentication

In CSAW, we want continuous authentication to be effortless, continuous, user-agnostic, quick, and secure (our design goals, as described in Section 2.4). We achieve the first two goals by design. CSAW requires no explicit input from the user and as long as the user is in (radio) proximity, i.e., the user's wristband can send data to the computer, CSAW continuously verifies the presence of the user; thus, CSAW does continuous authentication without any effort from the user. We conducted a laboratory study to evaluate the degree to which CSAW achieves the other three goals.

6.5.1 User study

We recruited twenty participants for our user study: ages 18 to 30; seven male and thirteen female; and eight from computer science (CS) background and twelve from non-CS background. We recruited participants using flyers posted across our college campus and online; our research protocol was approved by our campus Institutional Review Board (IRB). Participants took about 30 min to 40 min to complete the user study; they received \$10 as compensation.

The user study consisted of three experiments. The first experiment was designed to capture the users' hand movements as they interact with a desktop in *normal* use. Participants were instructed to imagine that they were in a public cafe and were asked to browse the web for 10 min. They were told that everything they typed would be logged and so were asked not to enter any sensitive information. Further, they were asked not to read any long articles or watch videos, as it would not provide much data for our study.

We designed our second experiment to collect user interaction data in a more controlled setting. Participants were asked to fill out a small web form, which required them to type, scroll, drag the mouse, click, and move the mouse; they were asked to fill this web form five times.

Our third experiment was designed to collect data to test a malicious adversarial case. For this experiment, we asked each participant to be a malicious adversary whose goal was to mimic the victim user's mouse-hand movements to the best of their abilities. The victim user (one of the researchers) filled out the same web form that the participants used in Experiment 2; thus, the participants were already accustomed to the task. We realize that a real adversary can be motivated and skilled enough to mimic users very well, compared to our participants. So we decided to assist the participants when they were performing the role

of a malicious adversary. To assist them in mimicking the victim, we made sure they had a good view of the screen and the victim's hand movement, we increased the cursor size, and the victim user gave verbal clues before he began an action. For example, the victim would say 'typing' before he began typing. He would say '2' when he was going to fill the question number 2 in the web form. The victim tried to use the same pace to fill out the web form for all participants, but reduced the pace for some participants when they were lagging too far behind. It should be noted that this experiment was intended to be favorable for the adversary!

Each participant performed, on average, about 192 Scrolling interactions, 293 Typing interactions, and 146 MKKM interactions in the three experiments. After these three experiments, we asked each participant to walk for a few minutes and to write on a paper, so we could collect data for walking and writing activities, because these are common activities for a user that steps away from the desktop. We use this data to evaluate how quickly CSAW can deauthenticate a user when she does one of these tasks while another user attempts to use her desktop.

We collected two types of data about participants' interaction with the desktop: *i*) inputs received by the OS through keyboard and mouse, captured by a script we wrote; and *ii*) the participant's hand movement, captured by a Shimmer worn by the participant on her dominant wrist.

We used iMacs for our participant study. Participants used an iMac with an Apple keyboard and Apple mouse with a scroll ball. We wrote a Python script that uses Apple's Cocoa APIs for OS X and captures all keyboard and mouse input events generated by the operating system when the participant provides input using those input devices. The captured input events were `KeyDown`, `KeyUp`, `MouseMove`, `ScrollWheel`, `LeftMouseDown`,

`LeftMouseUp`, and `LeftMouseDown`, which are generated when the participant presses and releases a key, moves the mouse, uses the scroll-wheel, presses and releases the left mouse button (left click), and drags the mouse, respectively.¹ For each keypress event the OS reports the time when it is pressed/released, the key value, whether the participant is holding the key down, and whether the participant is repeatedly pressing the key. For mouse-related events, the OS reports the time the event was generated, absolute coordinates of the mouse pointer on the screen, pointer displacement (in pixels) since the last mouse event, and scroll length (i.e., how much the participant scrolled the wheel). We log all this information, but in the current implementation of CSAW, we use only the time of event, event type, key value for keypress events, scroll duration, absolute mouse pointer coordinates, and mouse pointer displacement.

As for our study described in Section 6.3.1, we used the Shimmer to capture the participants' wrist movements: we asked each participant to wear the Shimmer on the wrist of his/her mouse-hand. We sampled the accelerometer and gyroscope sensors at 500 Hz; the sensor data was streamed to the desktop in real time during the experiment, where it was logged to a file. We synchronized the Shimmer and the desktop clock using the receive time of the Shimmer packets on the desktop. We calibrated the accelerometer and gyroscope sensors in all Shimmers prior to their use.

6.5.2 Usability

In CSAW's continuous authentication, the false-positive rate FPR is the fraction of all interactions where an unauthorized user is authenticated as an authorized user. Similarly, the false-negative rate (FNR) is the fraction of all interactions where an authorized user

¹Although our implementation is for MacOS X, the same kinds of information are available in Windows and Linux so our method should be easily portable to other systems.

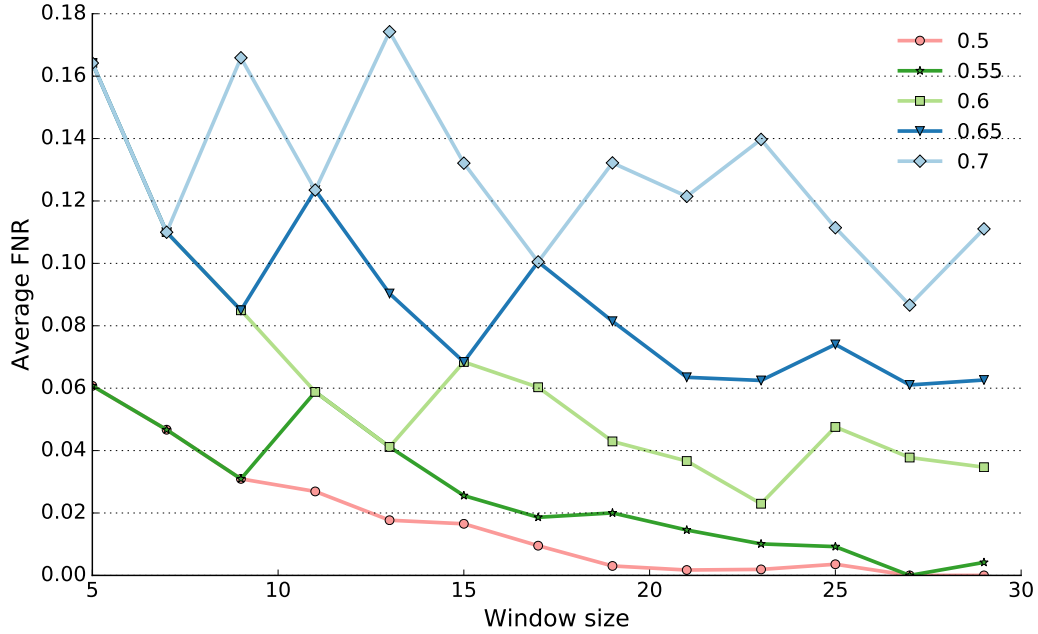


Figure 6.13: Average FNR vs. window size for different thresholds (0.5, 0.55, 0.6, 0.65 and 0.7). CSAW performs best in continuously authenticating users for window sizes larger than 20.

is misclassified as an unauthorized user. From the usability point of view, a low FNR is desirable. Figure 6.13 shows the average false-negative rate across all participants for different window sizes w and thresholds m .

As described above, CSAW classifies the desktop user as the wristband user by comparing the actual interaction sequence and the interaction sequence inferred by the classifier from wristband data. The comparison is performed over a given window size. Thus, we compute an FNR as the fraction of all windows where CSAW misclassified the authorized user as an unauthorized user. Each data point in Figure 6.13 is the average of the FNR across all participants for a given window size and threshold value.

The threshold parameter m indicates the fraction of interactions in a window that should match for CSAW to authenticate the user. Thus, the threshold value indicates how strict CSAW is when correlating interactions, and as expected the FNR is smaller for smaller

threshold values and it increases with threshold values. Window size is the number of interactions matched at a time to authenticate the user; a larger window size allows more interactions to be matched, so the FNR drops as the window size increases. CSAW performs best in terms of authenticating a user for window sizes greater than 20; thus, the more interactions a user provides while working on the desktop, the better CSAW performs. In terms of threshold values CSAW provides best FNR for 0.5 and 0.55, and reasonably well for threshold value of 0.6. A low threshold value improves usability but, as we show below, it reduces security, so we need to choose the trade-off carefully.

6.5.3 Security

We compute a FPR for each participant as the average FPR across all adversaries; each data point in the following FPR graphs is the average of these FPR across all participants. A high FPR indicates that we falsely authenticate an unauthorized user as the logged-in user, allowing him to access the logged-in user's account, which is undesirable. Thus, a low FPR is good from a security point of view. Window size is the number of interactions that CSAW is allowed to consider to issue the decision whether the current user is the same as the logged-in user. Thus, ideally we want a low FPR for a small window size.

Figures 6.14 and 6.15 show the average FPR when the adversary is using the desktop and the logged-in user is walking and writing, respectively, near the desktop. (Note the different y -axis scales.) As expected, the FPR is smaller for the higher thresholds. The FPR is low and drops quickly with respect to window size, when the user is walking compared to when she is writing, because the wrist movements while walking are very different to wrist movements when a user is using the desktop, whereas the wrist movements during writing are somewhat similar to desktop use.

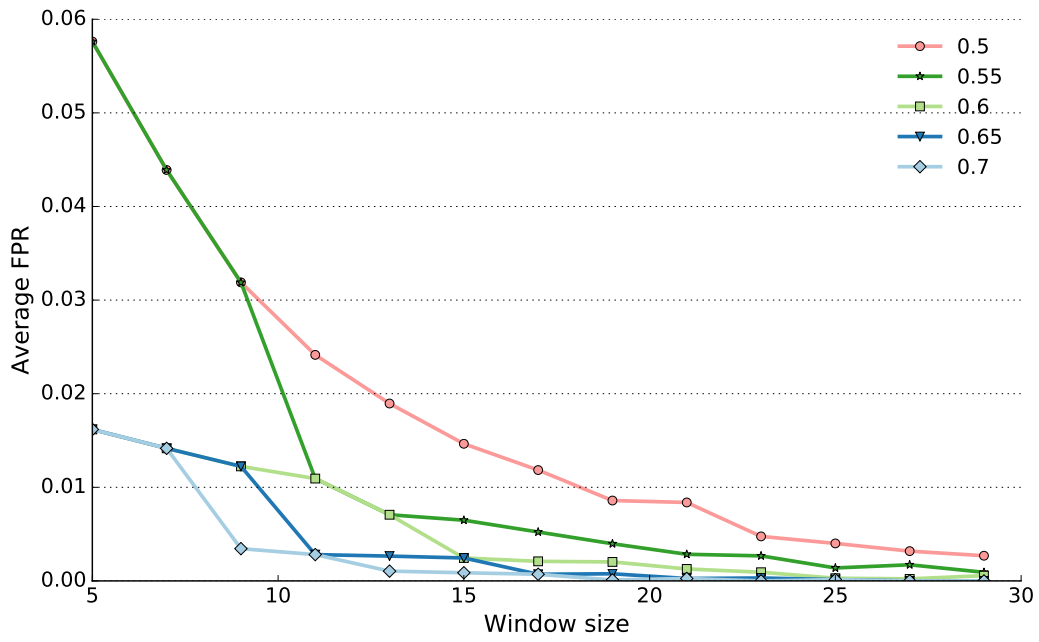


Figure 6.14: Average false-positive rate when the adversary is accessing the desktop while the logged-in user is walking nearby.

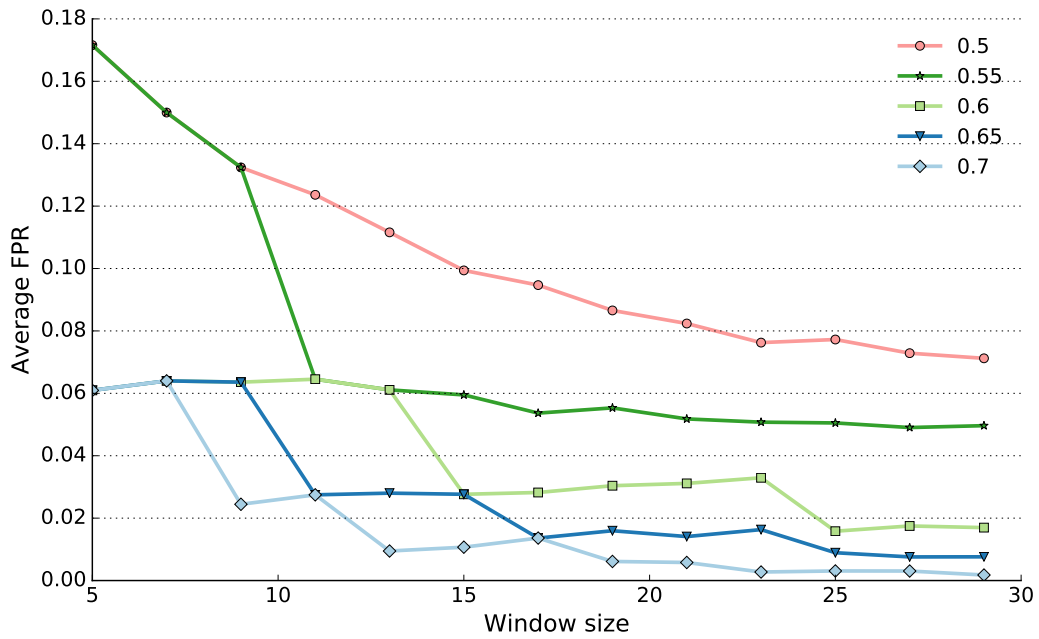


Figure 6.15: Average false-positive rate when the adversary is accessing the desktop while the logged-in user is writing nearby.

Figure 6.14 shows that the FPR is below 0.02 for thresholds 0.6 and above, even for short window sizes. The FPR in the user-writing case drops below 0.03 for threshold 0.6 for windows of size 15 or greater. Thus, CSAW performs reasonably well even if the user is performing an activity that is somewhat similar to working on a desktop in terms of hand movements.

In our third experiment, we imagine a malicious adversary: the user is logged-in on a desktop D_1 but steps aside to work on a nearby desktop D_2 , and the adversary starts using desktop D_1 while trying to mimic the user's hand movements and similar interactions. If the adversary succeeds in mimicking the user's hand movements while providing similar interactions to desktop D_1 , then CSAW will misclassify the adversary as the user and the adversary can continue using the desktop. In our experiment we asked the participants to be the malicious adversary and try to mimic a user (a researcher). Both the participant and researcher performed the same tasks (filling web forms), and the researcher's screen and hands were clearly visible to the participant. Figure 6.16 shows the false-positive rate for this case. The FPR rate drops below 0.04 for windows of size 15, and threshold 0.6 and above. Thus, even when the adversary and the user were performing the same task on nearby desktops, and the adversary was trying to mimic the user's actions, CSAW performed well in recognizing the adversary. Thus, CSAW should be able to recognize a change in user even in an environment where the previous user is working on a nearby desktop when a new user (adversary) steps in to use the unlocked first desktop.

6.5.4 User-agnostic

To evaluate how well CSAW performs when it is user-agnostic, we use leave-one-out cross-validation: for each participant we train the classifier using other participants' data.

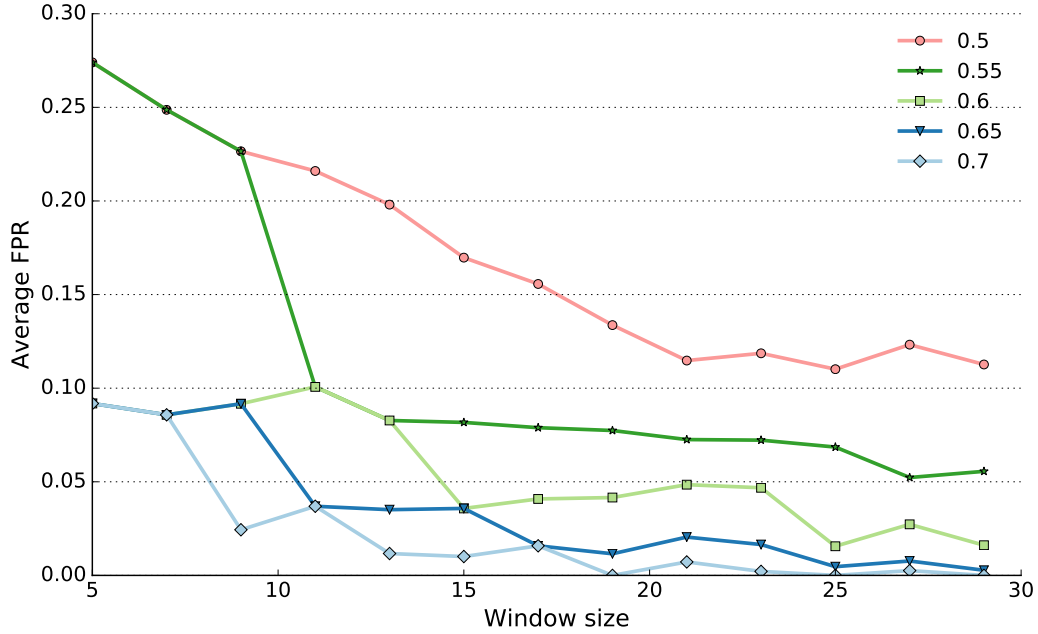


Figure 6.16: Average false-positive rate when the adversary is trying to access the user’s logged-in desktop by mimicking the user who is using a nearby desktop.

Table 6.2: Average false-negative rate (FNR) and false-positive rate (FPR) for different Window sizes (W); mean \pm standard deviation across all participants.

W	FPR ¹	FPR ²	FNR (all)	FNR (except P1)
5	0.016 \pm 0.012	0.061 \pm 0.064	0.164 \pm 0.155	0.140 \pm 0.118
13	0.007 \pm 0.008	0.061 \pm 0.088	0.041 \pm 0.077	0.026 \pm 0.038
21	0.001 \pm 0.002	0.031 \pm 0.057	0.037 \pm 0.096	0.017 \pm 0.044
29	0.001 \pm 0.001	0.017 \pm 0.035	0.035 \pm 0.094	0.015 \pm 0.034

¹ When the adversary is using the desktop while the user is walking nearby.

² When the adversary is using the desktop while the user is writing nearby.

Table 6.2 shows the average FNR and FPR across all participants, for threshold of 0.6 and four different window sizes; CSAW performed best for threshold of 0.6. For window size of 29 and threshold of 0.6, CSAW achieves reasonably low FNR (0.017 ± 0.035) and low FPR (0.035 ± 0.094) without learning participants’ individual desktop interaction behavior; thus, CSAW performs reasonably well when it is user-agnostic.

The high variability in FNR is because of participant P1: participant P1’s wrist movement during keyboard and mouse interaction were very different compared to the other participants,

so the classifier – trained with these other participants’ data – could not accurately classify P1’s interactions, which affects CSAW’s accuracy. This can be resolved by training the classifier on a larger population or training the classifier for these specific participants. If we exclude P1, we get low variability and even better FNR, as shown in the last column in Table 6.2.

Quickness

From the FPR and FNR results we found the parameters that give a reasonable tradeoff between usability and security with CSAW are window size of 21 and threshold of 0.6. We use these optimal parameters to evaluate *quickness* of CSAW in terms of the time CSAW takes to recognize an unauthorized user.

When the user changes (i.e., when the current user is different than the logged-in user), we want to identify the change immediately so we can prevent any accidental or intentional misuse of the logged-in user’s account. We define *quickness* as how ‘soon’ we can identify a changed user, where ‘soon’ can be measured in time or windows, where a window represents a fixed number of interactions. We use the ‘duration of attack success’ as a metric to evaluate how quickly CSAW detects an adversary and ‘duration of inappropriate lockout’ as a metric to evaluate how often CSAW will lock out an authorized user because it misclassified the user as an adversary. A smaller duration of attack success is better as it gives a smaller attack window for the adversary. On the other hand, it is desirable to have extended periods of time without inappropriate lockouts in order to improve usability.

Figure 6.17 shows the fraction of users that are recognized as authorized users by CSAW at time t for a grace period (g) of 1 and 2 windows. The figure shows the first instance in time when CSAW misclassifies the user as an adversary *and* takes action according to the

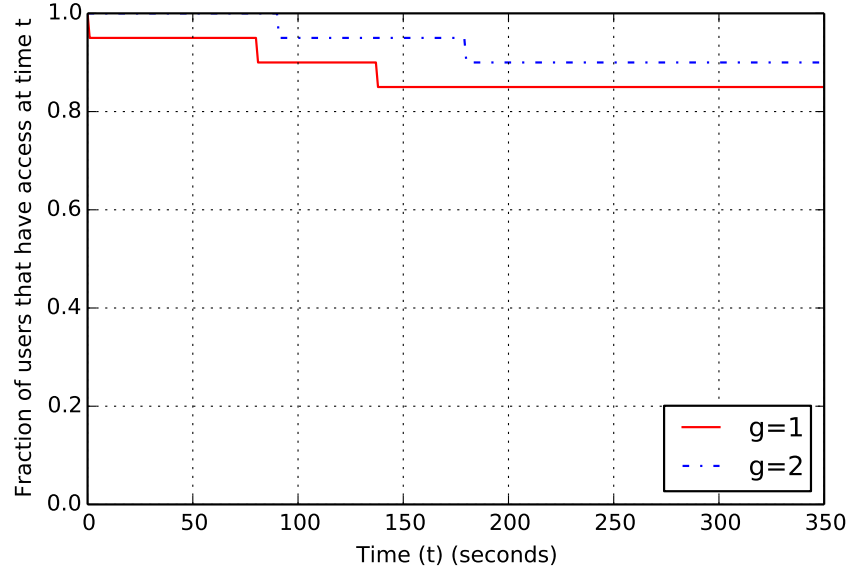


Figure 6.17: Fraction of authorized users that have access to the desktop at time t , for optimal window size = 21 and optimal threshold = 0.6.

system/user policy, which can be to lock the desktop or log out the user. For instance, 95 % of users were still recognized as authorized users at 50 s, or said another way, 5 % of users were misclassified by CSAW by time 50 s and may be required to re-authenticate themselves. For grace period of 1 window, CSAW correctly recognized 85 % of users throughout their session on the desktop. We can improve this number by increasing the grace period. As shown in the figure, for grace period of 2 windows, CSAW recognizes 90 % of users correctly throughout their use of the desktop.

Figure 6.18 shows the fraction of adversaries that are recognized as authorized users by CSAW at time t for grace periods (g) of 1 and 2. This graph shows how quickly CSAW can recognize an adversary and terminate his access to the logged-in user's account on the desktop. As shown in the figure, for grace period of one window at time $t = 0$, all adversaries have access to the desktop, but within 5 s only 40 % of adversaries have access – CSAW identified on average 60 % of adversaries as unauthorized users within 5 s, and by $t = 11$ s,

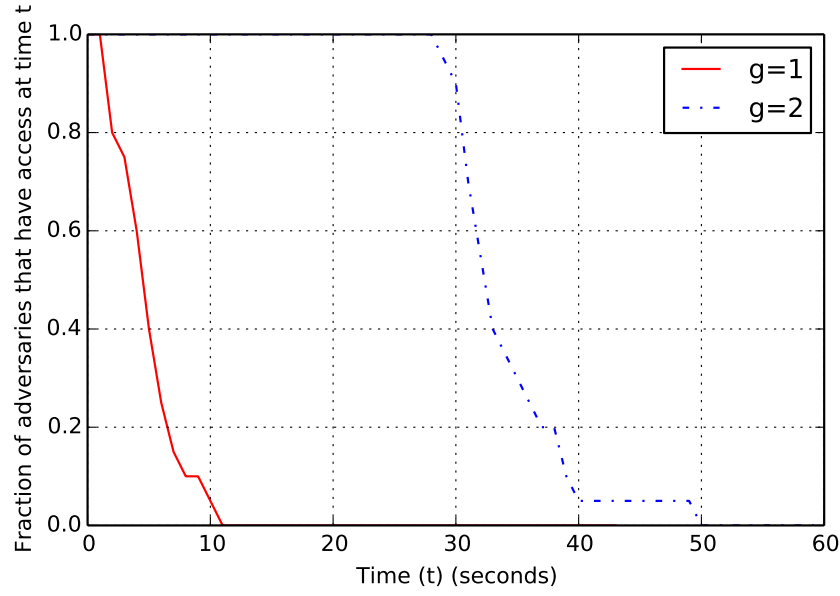


Figure 6.18: Fraction of adversaries that have access to the desktop at time t , for optimal window size = 21 and optimal threshold = 0.6.

CSAW identified all adversaries. A grace period of two windows improves usability of CSAW (see Figure 6.17), but it also increases the attack duration for adversaries; nonetheless CSAW still identified all adversaries within 50 s, much faster than typical deauthentication timer methods.

Figure 6.19 represents the previous graph (Figure 6.18) but in terms of windows instead of time, i.e., the fraction of the adversaries that have access to the desktop at the end of window w , where window size is 21 and threshold is 0.6. CSAW identified all adversaries for grace periods of 1 and 2 by the end of window 2 and 4, respectively. As we mentioned above, the window size is determined by the number of interactions (in this case 21), but interactions have variable duration; to compare with the previous figure, for all adversaries a duration of 2 windows was approximately 11 s.

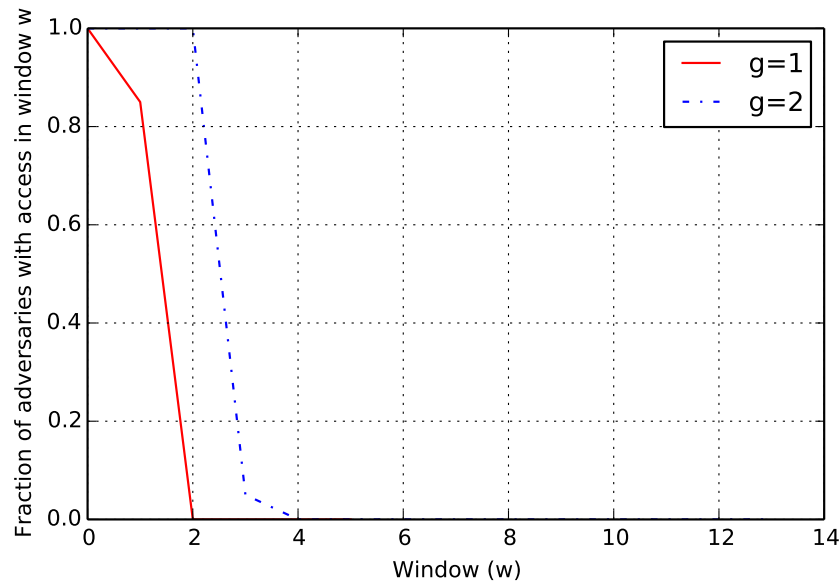


Figure 6.19: Fraction of adversaries that have access to the desktop at the end of window w , for optimal window size = 21 and optimal threshold = 0.6.

6.6 Deauthentication

When a user stops using the desktop, we need to deauthenticate the user to prevent any unauthorized access. To deauthenticate a user, we need to know *when* and *how* to deauthenticate the user.

When to deauthenticate? A user should be deauthenticated at the end of her session, i.e., when she stops using the target desktop. However, what marks the ‘end’ of a session is not always the same – it depends on the user preference and the context. At work, a user may prefer to end her session after 10 min of inactivity; at home, she may prefer to keep her session alive longer, say 30 min. At a bank, the IT security staff may want to end users’ sessions (and deauthenticate them) as soon as they step away from their desktops; in a hospital, however, the IT security staff may allow sessions to persist when the user is away from the desktop, but within specified distance range, because the clinicians and nurses’

workflow often requires them to step away from the desktop and then return to continue their session. Thus, when to deauthenticate a user is a matter of policy, chosen by the user and/or her employer. CSAW provides following four parameters to specify end of a session.

1. *Inactivity period*: The maximum time period without any desktop input (from keyboard or mouse) from a user.
2. *Distance*: The maximum distance between the user (her wristband) and the desktop during a session; distance is measured by the desktop using its wireless communication with the user's wristband (radio signal strength).
3. *Steps*: The maximum number of steps a user can take during a session; steps are measured by the user's wristband.
4. *Change in user*: If a different user (different than the logged-in user) starts using the target desktop; change in user is detected by CSAW through continuous authentication.

These parameters can be used to specify deauthentication policies that meet the needs of different contexts, applications on the desktop, and the users' preferences.

How to deauthenticate? Once we know when to deauthenticate a user, the desired response, i.e., *how* to deauthenticate, is also a matter of policy. The policy might dictate to lock the screen or to log-out the current user – a sudden deauthentication response. The policy may specify a graduated response, if the underlying authentication method provides a probability (rather than a binary value) indicating the confidence that the user is still using the desktop. As long as the probability remains high, the desktop operates normally. When the probability drops, the screen may dim, then darken, then lock, then log out – in each case offering the user an opportunity to take an action (increasingly complex, as the confidence

gets lower) to restore confidence in her authenticity. Such an approach can improve usability without necessarily lowering security.

With small modifications, CSAW can provide a probability of the user's authenticity rather than a binary decision. Recall that CSAW uses a threshold parameter m and a grace-period parameter g in making its decision; instead of outputting a binary decision, we could arrange for CSAW to output a probability intended to indicate its *confidence* that the user input is coming from the logged-in user. For other parameters (inactivity period, distance, and steps), CSAW can provide multiple outputs as the parameter gets close to the predefined threshold; for example, if the user can take at most 20 steps (steps threshold) during a session, CSAW can provide a output when the user takes 5, 10, 15 and 20 steps to provide gradual deauthentication response. Alternatively, CSAW could combine one or more of these parameters to provide an estimate of when the user has walked away from the computer; for example, if the user takes certain number of steps and is estimated to be certain distance away.

CSAW supports different deauthentication approaches and enables flexible deauthentication policies. How well users respond to these deauthentication approaches (and the security analysis of these approaches) is an open research problem that we hope to tackle in the future.

6.7 Discussion

In this section, we discuss important issues related to deploying CSAW in a real system, potential attacks, and extension to laptops.

6.7.1 Deployment issues

There are several important issues that need to be addressed to deploy CSAW in real world; we start with the limitations.

User handedness. For initial authentication, the user may wear the wristband in either hand, because the Tap-5x interaction is easy to perform with either hand. Continuous authentication, however, requires the user to wear the wristband on her mouse-hand (usually the dominant hand). This is a limitation for continuous authentication, because a user may prefer to wear a wristband on her non-dominant hand. People generally wear a wrist device on their non-dominant hand, either out of habit or because they use their dominant hand more than their other hand, and they wish to minimize interference with their task or avoid any damage to the wrist device. If the CSAW wristband is made sufficiently low-cost and small in form factor, we expect, people might wear it on their dominant hand; we often see people who wearing multiple wrist wristbands, wear them on both wrists.

Interactions for continuous authentication. CSAW's security in continuous authentication is based on the implicit assumption that users' interaction with desktops is interlaced with typing, scrolling, and MKKM interactions (see 6.4.1), and it is difficult to mimic all the interactions; in our evaluation we address the second half of the assumption (see 6.5.3). The assumption about interactions being interlaced comes from our observations of clinical use of desktop computers, which is primarily to access patient records and it involves all three interactions. This assumption holds true for many other users and use cases, but not always: some users prefer to use only the keyboard for most of their interaction with a desktop. If a user primarily provides only one type of interaction (e.g., only typing), CSAW cannot verify the user with high confidence.

Huhta et al. demonstrate this limitation through a selective mimicry attack, where an adversary mimics only one type of interaction (i.e., only typing or only scrolling) [35]; in their attack scenario, the victim user is using a desktop, but she forgot to log out of the target desktop (target for the adversary), and the adversary gets access to an unattended logged-in target desktop, and mimics only the user’s typing interactions by listening to (or observing) the user typing. Even with this caveat, CSAW offers better security than one-time authentication methods or timeout-based deauthentication, because the adversary would not be able to access the desktop indefinitely, as in these traditional methods; when the victim user stops using a desktop, the adversary would not be able to access the target desktop. Furthermore, if all the victim user’s desktops support CSAW, she can configure her wristband to allow access to only one desktop at a time, preventing any mimicry attacks.

Securing wristband to the user. CSAW does not verify a wristband wearer or provide resilience to theft – it relies on existing (or proposed) solutions. Apple Watch uses a PIN-based approach to verify the wearer [7]; in PICO, a token-based authentication method, the user is verified based on the other known wireless devices (PICO siblings; e.g., smartphone, smartwatch) that she carries with her [90]; and Cornelius et al. proposed a wristband can continuously recognize its wearer using biometrics [17]. Our preferred solution is the biometric-based verification, because it would make CSAW a truly memoryless authentication method – the user does not have to remember any password to use CSAW. From deployment perspective, however, the preferred approach might be to use a PIN or password to activate the wristband when the user wears it, along with a mechanism to detect when the wristband is taken off (to deactivate it); the user would have to remember and enter a password, but only once a day, when she first dons her wristband.

Energy consumption on wristband. The energy consumption of the accelerometer, the gyroscope, and the radio on the wristband determine the recharging interval, with longer times being preferable. In CSAW, to respond to a user's authentication intent quickly, the wristband is always sensing the user's wrist movement. Accelerometers that consume very low power (e.g., Analog Devices ADXL362, 3.0 microAmps at 400 Hz) could be used for continuous sensing. A gyroscope would use more power, but it can be kept turned off when not in use and enabled only when required; low-power accelerometers can be used to detect events that require the gyroscope (e.g., for authentication when Tap-5x is detected). Low-energy radios such as Bluetooth Low Energy and ANT+ can run for months on a button-sized battery and the radio could be used both as a test for rough proximity to the computer desktop and to wake the sensors when proximity is achieved. In our CSAW prototype, the wristband transmits raw sensor data to the desktop. Instead, the wristband could do computation locally to derive features and correlation points, and transmit those to the desktop, reducing the communication overhead. Together, with these optimizations, it should be possible to provide a battery life for a wristband for days, if not weeks.

Connectivity. One of the main limitations of CSAW is that Bluetooth, which is the most ubiquitous wireless protocol for personal area networks, was not designed for complex network topologies. As a result, pairing multiple desktops with a single wristband, or more generally, multiple devices with multiple wristbands, can be challenging. This limitation is not fundamental; there are other ultra-low power wireless protocols (such as ANT+ and TI's SimpleLink) that support many-to-many connectivity in high-density environments. The Bluetooth SIG Smart Mesh Working Group is currently working to develop low-energy Bluetooth smart-mesh networking support, which, we expect, would migrate into commercial

wristbands in the near future. Some wristbands support Wi-Fi as well, which meets the needs of CSAW; Wi-Fi radio is not low-power, but its energy consumption can be reduced using the optimizations described on page 144.

Keyboard variations. Keyboards vary in keystroke depth, key size, key shape, bounce, “clickiness”, and “feel”. We expect these variations would have an effect on the accelerations and rotations experienced at a user’s wrist. CSAW works with wrist motions that are already subtle and we expect keyboard variations can be easily handled, perhaps requiring a larger data set for pre-training the classifier.

Privacy leakage. Motion sensors in the wristband can leave private information: one can infer the user’s activity or even keystrokes when the user types [48, 99] from the motion sensors in the wristband. In our CSAW prototype, the wristband sends raw motion sensor data to the desktop, but in an implementation, the wristband would compute the necessary features (or correlation points) locally and send the features to the desktop. The transformed data would severally limit the inferences that can be drawn.

A different aspect of privacy is the issue of tracking users. Desktops have to identify wristbands in the vicinity during authentication, and in principle, desktops could track the whereabouts of individuals wearing those wristbands. However, CSAW does not make this problem worse than it currently is with ubiquitous personal-area-network technologies and systems; network managers already collect desktop login information in system logs.

6.7.2 Other attacks

In our security evaluation we considered opportunistic and malicious attacks where the adversary attempts to mimic the victim user. An adversary may collude with another

adversary or use sophisticated motion tracking to improve his chances of success; we discuss these attacks below.

Colluding attack on initial authentication. If there are two attackers one could induce a Tap-5x-like interaction from the user while the other executes Tap-5x on a target desktop at the same time. For instance, one adversary can shake hands with the user (specifically with the user's wristband hand) five times while his partner adversary performs Tap-5x, matching the timing of the shakes (either through practiced coordination or electronically relayed motion measurement); the adversary doing the Tap-5x could even be a robot. The precision of the timing required for this attack and the limits of human reaction time make this attack difficult to carry out, and the attackers are only likely to get one chance. A machine used to synchronize Tap-5x with the induced motions would be conspicuous in the context of the desktops being used, causing reports to authorities or direct interventions. Even if the adversary manages to succeed, with CSAW's continuous authentication, any momentary advantage the attackers gain is soon lost.

Automated attack. In principle, it is possible to automate mimicking: a camera-based motion tracking system monitors the victim user's hand movement and a specialized hardware provides interactions (Tap-5x, typing, scrolling, MKKM) to the target desktop. Serwadda et al. demonstrated an attack on touch-based continuous authentication for smartphones using a toy robot that provided programmed touch inputs to a smartphone [80]. In CSAW, a robot-based attack is difficult to carry out, because any automated interaction to the desktop should be timed with the user's wrist movement. Our view is that this sophisticated attack is stronger than what is necessary to beat passwords today. A video camera can be used to obtain passwords [73]. Also, if an adversary can plug a custom set of keyboard and mouse

into a desktop, he can potentially plug a hardware key logger (between the keyboard and the system) as well [81].

6.7.3 Extension to laptops

We focused on desktop authentication, because of the motivating clinical use cases and our choice to explore keyboard and mouse input interfaces for bilateral authentication. We expect the extension of CSAW to laptops to be straightforward for initial authentication and deauthentication, but non-trivial for continuous authentication. For initial authentication on laptops, we would have to account for the difference in desktop and laptop keyboards (see 6.7.1). For continuous authentication on laptops, we would have to train the interaction classifier with the scrolling and MKKM interactions with touchpad, which is straightforward. The challenging aspect is to account for the difference between mouse and touchpad: unlike mouse use, which is always with the one hand, touchpad could be used with either hand (the location of the touchpad makes it easy to use it with either hand); without knowing which hand (wristband or non-wristband) was used for touchpad input, we cannot reliably determine MKKM or scrolling interaction.

6.8 Summary

We propose an authentication method for desktop computers, CSAW, that provides initial authentication, continuous authentication, and automatic deauthentication. Our goal in CSAW was to develop a usable and secure authentication method that is quick and effortless to perform, captures users' intention to authenticate implicitly, does not depend on users' individual characteristics (e.g., any physical or behavioral biometric), and continues to verify the user while she has access to the desktop. We evaluated how well CSAW meets these

goals through three in-lab user studies with a total of forty-five participants. Our findings suggest that CSAW is indeed quick to perform (within 1 s to 2 s) and participants found the initial authentication to be effortless and unobtrusive. In our user studies, CSAW achieved an average (\pm standard deviation) FNR and FPR of 0.025 (\pm 0.061) and 0.018 (\pm 0.036), respectively; in the worst-case attack scenario that we evaluated, CSAW had an average FPR of 0.023 (\pm 0.063). For continuous authentication, CSAW had an average FNR and FPR of 0.037 (\pm 0.096) and 0.031 (\pm 0.057), respectively; in our user study, CSAW was able to identify all adversaries within 50 s. Leveraging continuous authentication and the wristband, CSAW enables flexible deauthentication policies; a user can be deauthenticated when the user steps away from the desktop, based on the user's distance from the desktop, when the user changes on the target desktop, or based on an inactivity period on the target desktop. In one of our user studies that was designed to understand users' perception about CSAW's usability, a majority of the participants rated CSAW as more usable than passwords, and several participants said they would be willing to wear a wristband if it offered CSAW-like utility (authentication). Based on our evaluation through in-lab user studies, CSAW shows promise for reducing users' burden of authentication to desktops, without compromising security.

7

Smartphone Authentication

Authentication and deauthentication are, unfortunately, manual processes that users repeat several times each day to lock and unlock their mobile devices. Mobile devices like smartphones and tablets provide access to a wide range of sensitive services and personal information (email, photos, social networks, bank transactions, health records, enterprise data, and more); an unlocked phone is vulnerable to snoop family, friends, co-workers, and passers-by. Furthermore, current phone authentication is an all-or-nothing decision; loaning

one's phone to a colleague or family member (e.g., to make a call or play a game) typically gives them full access to all the content in the phone and services reachable through the phone.¹

Current phone authentication methods are seen as slow or inconvenient, especially when users use their phones numerous times in a day. As a result, many users take risky shortcuts: about 44% of smartphone users do not use passcodes at all [50], and many others choose simple, easy-to-remember, and easy-to-type passwords. Smartphones and tablets need an *unobtrusive* and secure method for authentication (and deauthentication) that also allows for granular access control. Although this chapter focuses on smartphones, the techniques can also be applied to tablets; we use the term *phone* to refer to smartphones unless otherwise noted.

In Chapter 6 we presented CSAW for desktop computers; in this chapter we present CSAW for phones. CSAW allows a phone to passively and continuously verify that the phone is literally in the hands of its owner. CSAW is actually a fundamentally new service to applications and subsystems on the phone, a foundation for *initial authentication* (the phone unlocks when picked up by the owner), *deauthentication* (the phone locks when accessed by someone other than the owner), *limitation* (the phone allows guest access to many apps, but limits sensitive apps to the owner), and *delegation* (the owner can temporarily grant specific access to another specific person).

CSAW works by correlating the owner's wrist motion with phone motion and phone input and continuously producing a score indicating its confidence that the person holding (and using) the phone is indeed the owner. Unlike desktops, which are often shared, phones are primarily used by one user, its owner; we use the term *phone owner* to refer to the phone

¹Some applications enforce in-app authentication (e.g., corporate email clients) even when the phone is unlocked; CSAW provides a foundation for those kinds of application-specific policies.

user. (Recall, we defined *user* as an individual authorized to use the computer in Section 2.1.)

The rest of the chapter is organized as follows. Section 7.1 gives an overview of CSAW’s approach for smartphone authentication; Section 7.2 describes CSAW’s method to verify the user; Section 7.3 presents the user study conducted to evaluate CSAW and the evaluation based on the user study; Section 7.4 discusses some limitations of CSAW on smartphones; and Section 7.5 concludes with a summary.

7.1 Overview

CSAW’s approach for phone authentication is inspired by the fact that smartphones and tablets are *hand-held* mobile devices. We interact with our phones using our hands: we provide touchscreen input (taps and swipes), we pick them up, we carry them around. The phone can monitor its own motion with its internal accelerometer and gyroscope sensors, and can observe touchscreen inputs (taps and swipes) as recorded by the phone OS. The phone’s motion and touchscreen inputs should correlate with the owner’s hand movements (measured by the motion sensors on the wristband), verifying that the owner is in fact the person holding (or using) the phone. Indeed, CSAW can passively continue to monitor the correlation and deauthenticate the owner when the wrist motion no longer matches phone input or motion.

The underlying CSAW goal is to provide a continuous quantitative estimate of the confidence that the individual using the phone is in fact the phone’s owner, assuming that the phone’s owner is wearing the owner’s wristband. CSAW uses bilateral approach for user authentication; for each user-phone interaction, it correlates two different observations by two distinct devices: 1) an observation derived from motion data from sensors in the wristband; and 2) an observation derived from motion sensors and touchscreen in the phone. CSAW

feeds these observations to an agent that estimates the likelihood that the two observations correspond to the same interaction. The agent then combines the likelihood numbers from recent interactions to provide its *confidence metric* to the phone operating system and, in some cases, to applications. Depending on the use-case and on policies set by the owner (or owner's employer), the system (or application) can use the confidence metric to take actions (e.g., to unlock the phone when picked up by its owner, or restrict access to certain applications according to who is using the phone).

Before we dig deeper into the specific CSAW approach in Section 7.2, let us look at how a phone could use the CSAW confidence metric to support initial authentication, deauthentication, limitation, and delegation.

Initial authentication. CSAW's interaction correlations could be used by an agent to implement fast initial authentication, leveraging natural interactions before the individual uses the phone or an application. Authentication could start even before the screen is on – for example, while the individual is getting her phone out of her pocket or while she is walking with the phone in hand – so she need only press the home button to turn on the screen and the authentication is complete. CSAW could also be combined with a fingerprint reader, or a mechanism such as Smart Lock from Android, to make a strong multi-factor authentication system.

Deauthentication via continuous authentication. Our approach for automatic deauthentication is a combination of continuous authentication and timeouts. Individuals using CSAW for automatic deauthentication can comfortably set a long timeout interval, knowing that if anyone else attempts to use their phone before the timeout, the CSAW agent will recognize that the individual using the phone is not the phone's owner and then lock the phone.

Limitation and access management. Consider a smartphone owner who lends her phone to a child to play games, or who lends her phone to a friend to search the Internet for a recipe. She may worry that app notifications could display personal information or that the borrower may launch an app providing access to sensitive systems at her place of work. CSAW's agent could support a notification engine that presents app notifications only when the owner is using the phone, or support an OS home-screen app launcher to limit which applications can be used when a guest (non-owner) is using the phone.

Delegation to other CSAW users. Consider again a smartphone owner who wants to lend her phone to a spouse or a trusted co-worker to allow them to access specific applications or view specific photo albums, without allowing them full access to the phone. The CSAW agent could provide her an interface to delegate access to that trusted individual, essentially, introducing her phone to that individual's wristband (and thus to that individual's identity) so that the delegated individual may use her phone in the future for certain approved purposes. Although this mechanism requires thoughtful attention to the user interface (and such a design is outside the scope of this dissertation) the CSAW system could be easily extended to support such a use case. CSAW would recognize multiple wristbands and receive motion data from those that are present, generating a confidence score for each one; the person whose wristband data generates the highest confidence score is deemed to be the current user of the phone, and that information could drive the phone's access-control decisions.

7.1.1 Challenges

CSAW faces two main challenges in its effort to determine whether the phone is in use by its owner.

Two hands, one wristband: CSAW correlates the wristband and the phone’s motion to verify that the phone owner is using the phone. If the owner holds the phone in her non-wristband-hand, CSAW cannot use motion correlation. We address this challenge by first identifying this use case using the wristband and the phone orientation. Second, if the owner provides inputs with the wristband hand (while holding the phone in the other hand), CSAW authenticates the user by correlating the wristband and phone’s motion *only* when the user provides taps and swipes on the phone. When the user holds and provides input to the phone using only the non-wristband hand, CSAW cannot authenticate the user for that interaction duration.

Temporal correlation: Authentication methods based on gestures also use motion correlation, but they correlate the current gesture with the user’s template gesture collected in the past. These methods are vulnerable to someone copying the user’s gesture style [47]. To increase the difficulty of such mimicking attack CSAW uses temporal correlation, which means that the adversary has to mimic the owner’s *current* wrist motion *in real-time*. Enforcing temporal correlation when comparing two motions that could be loosely coupled is challenging. We address this challenge by careful selection of features for our machine learning classification models.

7.2 Method

The CSAW agent monitors wrist motion, phone motion, and phone inputs so it can determine whether they are correlated and produce a summary metric we call the *confidence score*, a value between 0 and 1 that indicates the agent’s confidence that the individual using the phone is indeed the wristband-wearing owner. Specifically, it produces fresh scores

every 1-2 seconds: an instantaneous estimate $C(t)$ based on the latest data at time t , and an exponentially weighted moving average $\overline{C}(t)$ that smoothes recent scores:

$$\overline{C}(t) = (1 - \alpha)C(t) + \alpha\overline{C}(t - 1) \quad (7.1)$$

where the factor α weights the contribution of the past confidence score. As described above, application and system policies can use these metrics to drive authentication-related decisions. In this section, we describe the structure of the CSAW agent and how it produces this periodic confidence score.

Figure 7.1 depicts CSAW’s architecture and its modules. CSAW receives a steady stream of motion data from the phone (P_m) and the wristband (W_m), and touchscreen input data from the phone (P_i). These data flows are segmented into windows and examined by three modules, described below: the “grip detector” determines how the user holds the phone, the Motion-to-Motion Correlator (M2MC) correlates phone and wristband motion, and the Motion-to-Input Correlator (M2IC) correlates phone input and wristband motion. The resulting correlation metrics are considered by the scoring engine that actually computes $C(t)$.

Although CSAW outputs a confidence value, frequently, not all of its modules need be active continuously. When the wristband is not present, perhaps because the owner has stepped out of range of the phone, there is no wristband data; the correlation modules simply output 0 (not correlated) and that drives the confidence score $C(t)$ to 0 (lowest confidence). When the wristband is present but the phone is experiencing no input or motion, perhaps sitting on a table or in a bag near the owner, all the modules are inactive (to save energy) and the confidence score is again $C(t) = 0$. Indeed, CSAW calculates the correlation

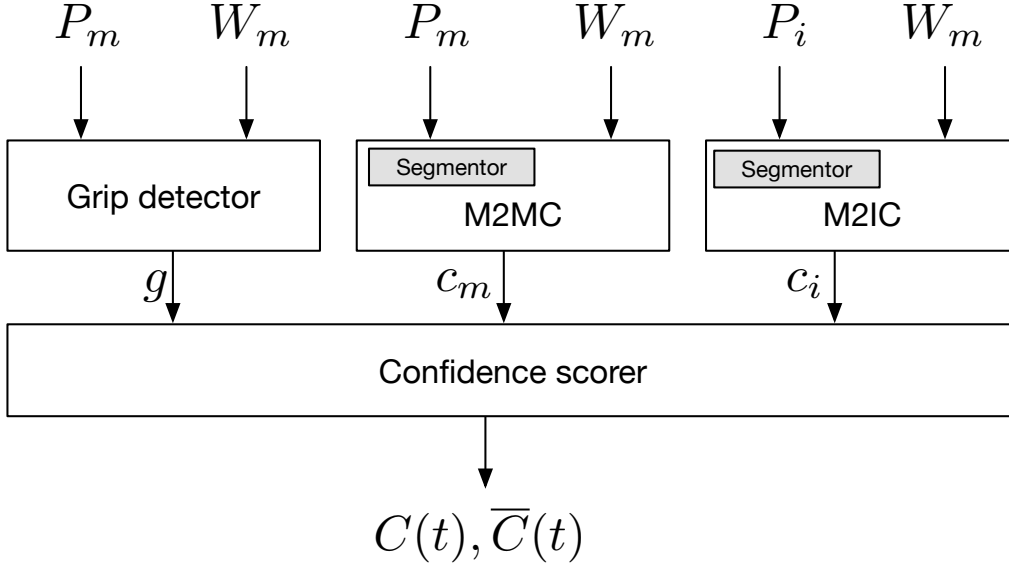


Figure 7.1: CSAW architecture.

between the wristband motion and the phone motion/inputs *only* when there is a user-phone interaction, that is, any action by the user that provides input to the phone or moves the phone; examples of user-phone interaction include the user picking up her phone or sending a text message on her phone. As soon as phone input or motion is detected, the wristband is instructed to start sending motion data, and the correlation modules become active. We discuss other energy optimizations in Section 7.4.

The data stream P_m and W_m is a series of sensor data samples of the form

$$s = (t, ax, ay, az, gx, gy, gz)$$

where t is the time when the sample was collected, and ax, ay, az and gx, gy, gz are the values from accelerometer and gyroscope sensors along their x, y, z axes, respectively. The stream of phone touch events P_i contains series of samples of the form

$$p = (t, id, x, y)$$

where t is the time when the sample was collected, id is the unique ID assigned by the phone OS to identify touch events performed by a single tracker (or finger) in order to distinguish touch events in a multi-touch input, and x and y are the x - and y -coordinates on the touchscreen where the user touched the screen respectively.² A *tap* is a series of touch events with the same tracking ID (id) but different timestamp (t) and position (x, y). Similarly, a *swipe* is also a series of touch events with the same id and with different t , x , and y , but a swipe is a longer touch interaction than tap, so it contains more samples than a tap.

7.2.1 Grip detector

Phones can be used with one hand or with both hands. When the phone is being used the grip module recognizes the *grip*, i.e., which hand is holding the phone and which hand is providing input. We describe a grip using two characters XY, where X is the hand holding the phone and Y is the hand providing input. The owner can hold the phone (or provide input) with the wristband-hand (W), the non-wristband-hand (N), or with both hands (B); if the phone is not held in either hand, we denote it with U (unknown). Thus, X and Y are values from the set {W, N, B, U}.

Motion-to-motion correlation (correlation between the wristband motion and the phone motion) is stronger when the wristband and the phone are tightly coupled, i.e., when the owner is holding the phone with the wristband-hand (grip WW, WN, or BB), but weak when the owner is holding the phone with the other hand (grip NW or NN). Motion-to-input correlation (correlation between the wristband motion and the phone's touch inputs) is stronger if the wristband-hand is used to provide the touch inputs. Knowing the grip during

²We used an Android phone in our experiments and these values are what the Android OS reports; another phone OS may report touch events slightly differently.



Figure 7.2: Wristband orientation in three grips. From left to right: NW (phone held in non-wristband-hand and input with wristband-hand), BB (phone held with both hands), and WN (phone held in wristband-hand and input with non-wristband-hand).

a user-phone interaction helps CSAW use the appropriate correlation method to improve the accuracy of the confidence score.

When the phone is in use, CSAW identifies the grip using the orientation of the wristband relative to the phone's orientation. The wristband orientation (relative to the phone's orientation) is different when 1) the wristband-hand is only used to hold the phone (input given by non-wristband-hand), 2) when the wristband-hand is used to hold the phone and provide input, and 3) when the wristband-hand is only used to provide input (phone held with the non-wristband-hand). Assuming the wristband is worn with its face on the outside of the wrist, and the phone is relatively horizontal, the above cases have the wristband face 1) horizontal facing down, 2) slightly vertical facing outwards, and 3) horizontal facing upwards, respectively, as shown in the Figure 7.2.

We use the difference in acceleration of the wristband and the phone along their z -axis to determine their relative orientation to each other and the grip. When the wristband and the phone are facing in the same direction (NW in Figure 7.2) the difference is close to

zero, and difference is maximum when they are facing in the opposite direction (WN in Figure 7.2); for BB and similar grips if the wristband’s z -axis is aligned at 90° angle to the phone, the difference would be about half the maximum difference. Thus, these grips can be differentiated using thresholds on the wristband orientation relative to the phone; we determine the thresholds using the data we collected from our user study.

This module produces an output indicating the grip only when the phone is in use. The output at time t is a two-character string $g(t)$ representing the grip: BB, WN, NW, or UU; UU (unknown) implies this module could not determine the grip. CSAW treats WW and BB as the same grip, because in both these grips wristband-hand is holding the phone and their relative orientations are similar. This module cannot determine grips that do not involve the wristband-hand (NN), such as grips where the phone is resting on a surface and input is provided by the non-wristband-hand.

7.2.2 M2MC

The Motion-to-Motion Correlator (M2MC) module correlates the wristband motion with the phone motion to determine whether the phone is held by the user wearing the wristband. When the phone is in motion or in use, this module receives two continuous streams of motion data from the wristband (W_m) and the phone (P_m). The input data streams are segmented using sliding windows of size w_m with overlap fraction o_m ; CSAW uses $o_m = 0.5$ and $w_m = 2$ s when the phone is locked and $w_m = 4$ s when it is unlocked. We use a shorter window when the phone is locked for a quick initial authentication decision. For a window ending at time t , the module outputs a correlation score $c_m(t)$, between 0 and 1 that indicates the quality of motion correlation, with 1 indicating perfect correlation.

Features. For each segmented window, M2MC computes a correlation feature vector F_c with 256 features from time and frequency domains, as we explain next. We begin with *mean, standard deviation, mean value crossing rate, variation, interquartile range, median, mean absolute deviation, skew, kurtosis, power, energy, and peak-to-peak amplitude*, each a statistical representation of a signal. Then we compare two signals by computing two numbers for each of those 12 statistics – absolute difference and relative difference (ratio) – resulting in 24 features. These features, however, compare aggregate statistics of the two signals without examining whether they vary the same way with time. To those 24 *non-temporal* features we add eight *temporal* features, as follows. Two (*cross-correlation* and *correlation-coefficient*) measure similarity between two signals and how they vary together in the time domain, and a third (*coherence*) measures similarity and variance in the frequency domain. Five more features select the two highest peaks in both signals and compare their corresponding peak timestamps, amplitudes, and inter-peak times. Thus, for any two signals we compute 32 features. Since there are four signals (x , y , z and magnitude) each from the accelerometer and gyroscope, the final feature vector F_c has $256 = 4 \times 2 \times 32$ features.

To determine the correlation score, M2MC uses a classification model (classifier) that estimates the probability that a given feature vector F_c represents two motions that correlate. We trained the classifier with feature vectors labeled as *positive* (correlated) and *negative* (not correlated). We generated positive and negative samples to train the classifier using the data from our user study: positive samples were generated using wrist-motion data and phone motion data from the same participant and negative samples were generated using wrist-worn data from one participant and phone-motion data from another participant. For a given feature vector, the classifier computes probability estimates for the two labels; M2MC outputs the classifier’s probability estimate for the positive label as its correlation score

$c_m(t)$.

7.2.3 M2IC

The Motion-to-Input Correlator module correlates wristband motion with phone touch inputs to determine whether the touch inputs are given by the owner using her wristband-hand. It outputs a correlation score $c_i(t)$, a value between 0 and 1 indicating quality of correlation between the wrist motion and the phone inputs in the window starting at time t .

This module receives a stream of wrist motion data (W_m) and a stream of phone touch events (P_i). We segment the input data streams (W_m and P_i) based on touch interactions. Each window of data represents exactly one touch interaction (tap or swipe). For a touch interaction ending at time t we extract the corresponding motion data from the wristband (using the start and end time of the touch interaction). We then compute a feature vector from the wristband data, using the same 12 standard statistics (*mean, standard deviation, mean value crossing rate, variation, interquartile range, median, mean absolute deviation, skew, kurtosis, power, energy, and peak-to-peak amplitude*) of the accelerometer and gyroscope magnitudes. The result is a 12-feature vector F_t .

We then use a two-tier classification approach to correlate wrist motion with touch inputs. In the first tier, we determine whether F_t is a touch interaction (tap or swipe) using a binary classifier that outputs a label 1 if the wrist motion is like a touch interaction, otherwise 0; we trained this classifier using wrist-motion data from our user study, choosing participants' wrist-motion data when they were performing a tap or swipe as a label 1 and wrist-motion data from other activities (such as walking, wrist stationary, and typing on computer) as label 0. In the second tier, we use interaction-specific classifiers to identify the likelihood that F_t is indeed the touch interaction (tap or swipe) performed by the user; these classifiers

were trained using the wrist-motion data of our participants when they performed taps and swipes in our user study.³ M2IC then outputs the likelihood score from the second tier as its correlation score $c_i(t)$; if the wrist-motion was determined to be not touch interaction in the first-tier classification, the module outputs 0.

7.2.4 Confidence Scorer

This module periodically outputs the confidence score $C(t)$ at time t . To generate the confidence score, the module uses the correlation scores from the M2MC module $c_m(t)$ and the M2IC module $c_i(t)$, and the grip $g(t)$ from the grip module. CSAW intentionally favors M2MC, because M2MC computes correlation over more data, its output is more frequent, and it does not depend on the user's touch events. Indeed, CSAW uses M2IC only when output of M2MC is not reliable, i.e., when the phone is not held with the wristband-hand. In short,

$$C(t) = \begin{cases} c_i(t) & \text{if } g(t) \in \{\text{NW}, \text{UW}\} \\ c_m(t) & \text{otherwise} \end{cases}$$

The confidence-scoring module could be generalized to use a weighted average $C(t) = ac_m(t) + bc_i(t)$ instead, where a and b are weights ($a + b = 1$) determined by the grip g and other information. When the phone is in use, this module outputs confidence score once every two seconds (window size of four seconds with 50% window overlap) and the score is generated using data from only the current window. This instantaneous confidence score may be too reactive to base authentication decisions for some applications; such applications may prefer a more stable confidence score that also accounts for past scores, so this module also outputs an Exponentially Weighted Moving Average (EWMA) as in Equation 7.1.

³Both classifiers can be pre-trained and integrated into M2IC; they need not be trained for a particular user. It may be possible, however, to enhance these classifiers to learn the owner's particular style and improve accuracy over time.

7.2.5 Policy Decisions

The CSAW agent does not itself make any authentication-policy decisions, or take any actions relating to such decisions. Its role is simply to produce the confidence score as an input to applications or subsystems that need to make such decisions and would benefit from knowledge that the phone is indeed in the hands of its owner. As noted in Section 7.2, CSAW’s confidence metric may be used for decisions relating to initial authentication, deauthentication, limitation, or delegation. Depending on policies set by the owner (or owner’s employer), the system (or application) can use the confidence metric to take actions (e.g., to unlock the phone when picked up by its owner, or restrict access to certain applications or websites according to who is using the phone). Perhaps the simplest approach is a threshold policy: at time t , if $C(t) \geq \tau_c$ for some pre-defined threshold τ_c , proceed as if the user is indeed the owner. Conversely, when $C(t) < \tau_c$, assume that an imposter is using the phone and take protective action. This simple-minded approach may be too reactive, as $C(t)$ changes every 2 seconds, and false negatives (inappropriately low C when the owner is in fact the user) will make the phone unusable. The EWMA may be more reliable: proceed if $\overline{C}(t) \geq \tau_c$. Other options are possible, such as voting ($C(t) \geq \tau_c$ for at least k out of n recent values), trending (a non-decreasing slope of $C(t)$ values over a recent window), and so forth. We evaluate the simple-minded approach for decision making; a full exploration of policies, protective actions, and their use of the confidence score is out of scope for this dissertation.

7.2.6 Confidence booster

CSAW *can* help with another important category: second-chance actions. In situations when the confidence value derived from natural user-phone interactions is low, the system could

ask the owner to perform some explicit simple actions to improve confidence. For example, in a system using CSAW for initial authentication, if the phone fails to unlock due to low confidence in the M2MC correlation during a pick-up maneuver, it could display a notice asking the user to place the phone in their wristband hand and quickly rotate their hand (and phone) two or three times; CSAW can easily correlate these motions (as shown in Section 7.3). Alternately, after asking the user to place the phone in their wristband hand, the phone could vibrate and CSAW could examine the wristband accelerometer data for a vibration signal.

7.3 Evaluation

We evaluate CSAW’s effectiveness through a laboratory user study. Based on our results from our study, we discuss the security and usability benefits that CSAW offers.

7.3.1 Data collection

The user study was designed to capture participant’s phone usage data, i.e., their user-phone interactions. We recruited sixteen participants through flyers posted across our campus. Participants took about 30 min to 40 min to complete the study; they received \$10 as compensation. The study was approved by our university’s IRB.

It is difficult to capture people’s natural user-phone interactions in controlled experiments in a laboratory. In our experience, asking participants to simply ‘use their phone’ yields limited user-phone interaction, as they are likely to play a game or casually browse website(s) to pass the time. When people interact with their phones, they do it to perform a task and the user-phone interactions differ according to the task: consider user-phone interaction while reading news, checking and replying to emails, playing games, or typing. We wanted to

capture participants’ natural user-phone interaction for these different tasks, so we gave them six well-defined tasks, but did not instruct them on how to use the phone:

1. *Pick-up*: Pick up the phone from the desk, unlock it, take a picture, lock it, and put it back on the desk. Repeat this task 5 times with grip WW.
2. *Search*: Do 5 search queries with grip BB.
3. *Email*: Read emails and act on them (reply, delete, or archive) according to the action given at the end of each email with grip WW.
4. *News*: Read/skim through articles on Flipboard for 3 min with grip NW.
5. *Typing*: Type three given sentences on the phone, each with a different grip (WW, NW, and BB).
6. *Rotate*: Pick up the phone and do a ‘rotate’ action (rotate it clockwise, and then counter-clockwise, bringing it to the starting position). Repeat the task 5 times with grip WW.

To facilitate tasks for participants on the provided Android phone, we created dummy user accounts for apps and services (Gmail and Flipboard) that participants used during the study. The Email and the News tasks are meaningful if the user account has unread emails and news feeds to follow. We initialized the email account by sending emails (sampled from the Enron email dataset [25]) to the dummy account; each email had a sentence at the end that instructed the participant whether to reply, delete, archive, or simply ignore the email. We initialized the Flipboard accounts by subscribing to some news sources. To facilitate the Search task, we suggested search queries (chosen a dataset of queries performed by users) to participants: we printed twenty search queries on a sheet of paper and participants were free

to use these search queries, if they wished; 13 out of 16 participants chose queries from the provided list. Each participant provided about 10 min of continuous phone use data (search, email, news, typing), 5 pick up actions, and 5 rotate actions.

Participants wore a Shimmer [82] on their dominant wrist during the experiment and they were given an Android phone for the duration of the experiment. During the experiment, we captured wrist motion (acceleration and angular velocity from the Shimmer), phone motion (acceleration and angular velocity from the phone’s sensors), and phone touch events. The motion sensor data on the phone was captured at 200 Hz frequency. We captured data on the Shimmer at 500 Hz, but down-sampled it to 200 Hz, to match the sampling frequency on the phone. We used the Android `adb` utility to capture touch events on the phone.

7.3.2 Effectiveness

This section evaluates the degree to which CSAW was effective, that is, it produced accurate confidence scores. Since all of our aforementioned scenarios (Section 7.1) use the confidence score as input into an authentication policy decision, one way to evaluate the quality of the confidence score is to evaluate the quality of some authentication decisions. In such decisions, a ‘positive’ decision means that the policy decided that the individual using the phone is the owner and access should be permitted; a ‘negative’ decision means the opposite. From the user study data, we generated positive samples (a segmented window with the wristband data and phone data from the same participant) and negative samples (a segmented window with the wristband data and the phone data from different participants); we use these test samples to compute false-positive rate (FPR) and false-negative rate (FNR). Because there was a much larger number of negative samples than positive samples available for testing, we also report balanced accuracy (BAC) metric.

For our evaluation, we consider the simplest possible authentication policy, which makes a new decision each time CSAW provides a new confidence score $0 \leq C(t) \leq 1$. This policy is a simple threshold comparison and classifies according to the immediate confidence score: positive if $C > 0.5$ and negative if $C \leq 0.5$.

Initial authentication. With CSAW one should be able to pick up a phone and achieve initial authentication, so we used the data from the ‘pick up’ task in the user study. In this scenario, the phone is locked and sitting on a table; when a person picks up the phone, the phone should unlock for the owner (no false negatives) and not the imposter (no false positives). When the phone is locked there is no input, so CSAW ignores the grip and M2IC and produces a confidence score from M2MC alone. Our experiments show that M2MC was more than 99% accurate. The average FNR was 0.008 ± 0.010 , indicating high usability. The FPR was at or near zero, indicating high security, but let’s look closer. Consider situations where an imposter picks up the phone while the user is working on a PC, using another phone, picking up another object, walking, or stationary; Table 7.1 shows the average FPR and BAC for each of these cases, presenting the mean and standard deviation over a 10-fold cross validation on our dataset. In nearly every case, there were virtually no false positives and near-perfect accuracy. The hardest case occurred when the imposter and the user both performed the same type of action (pick up); nonetheless, CSAW performed well with a low FPR of 0.017 ± 0.012 .

Confidence booster. Although the False Negative Rate for initial authentication was below 1%, every such failure is an annoyance to the owner. As discussed in Section 7.2.6, CSAW can improve usability in such cases by asking the user to perform a simple explicit action to boost its confidence in the owner’s presence as the holder of the phone. We thus evaluated

Table 7.1: Average false-positive rate (FPR) and balanced accuracy (BAC) across all participants for pickup interaction; mean \pm standard-deviation.

Activity	FPR	BAC
PC use	0.000 \pm 0.000	0.999 \pm 0.003
Phone use	0.000 \pm 0.000	0.999 \pm 0.003
Pick up	0.017 \pm 0.012	0.990 \pm 0.007
Walking	0.000 \pm 0.000	0.999 \pm 0.003
Stationary	0.000 \pm 0.000	0.999 \pm 0.003

Table 7.2: Average false-positive rate (FPR) and balanced accuracy (BAC) across all participants for rotate interaction; mean \pm standard-deviation.

Activity	FPR	BAC
PC use	0.000 \pm 0.000	0.996 \pm 0.005
Phone use	0.000 \pm 0.000	0.996 \pm 0.005
Rotate	0.009 \pm 0.003	0.991 \pm 0.005
Stationary	0.000 \pm 0.000	0.996 \pm 0.005
Walking	0.002 \pm 0.003	0.995 \pm 0.005

CSAW’s performance for a rotate action, in which the user holds the phone in her wristband-hand and rotates her forearm so that the wristband and the phone both experience the same rotate motion. We used the data from the ‘rotate’ action in our user study, and form the test dataset for different cases as discussed above. Because the phone is expected to be in the wristband-hand (grip WN), this confidence booster uses only M2MC. The average FNR from a 10-fold cross-validation was 0.002 (\pm 0.006); Table 7.2 shows the average FPR and BAC for different cases. Among the different activities, ‘rotate’ represents the hard case because the user and imposter both performed the same action; the average FPR even for this hard case was low 0.009 (\pm 0.003). Performing a rotate action takes less than one second, so this can be a quicker way for the phone owner to verify herself than entering a passcode or giving a fingerprint; if the user is already holding the phone with wristband-hand, she does not even have to change the phone grip.

Continuous authentication. Finally, we evaluated CSAW’s performance for continuous authentication; in this case, the full CSAW infrastructure is relevant (M2MC, M2IC, and the grip detector), because the user may hold the phone in either (or both) hands, provide input with either (or both) hands, and switch grips during use. We drew samples from the ‘phone use’ tasks that participants performed in our user study. In this case, the imposter has received (or taken) the unlocked phone from its owner, and attempts to continue using the phone while its owner performs some other activity. (The ‘imposter’ may be a friend or family member, and not necessarily a malicious stranger; recall that continuous authentication is the foundation for limitation and delegation as well as deauthentication.) Table 7.3 lists these other activities: the owner was stationary, walking, using a PC, using another phone, or doing the same task as the imposter. The results show a relatively high 2-7% FPR. Keep in mind, though, that these decisions are made once every two seconds, so the chances are slim that an imposter can maintain a consistent series of positive outcomes for more than a few seconds. The average FNR for continuous authentication was 0.024 (± 0.007), which appears large enough to be unusable, but this is also produced every two seconds. Furthermore, this result applies to the simple-minded policy defined earlier in this section; we anticipate that a practical policy would use the EWMA $\overline{C}(t)$ or other smoothed version of the confidence score as the basis for its decisions. For instance, Figure 7.3 shows instantaneous confidence score $C(t)$ for participant P1 and EWMA $\overline{C}(t)$ with $\alpha = 0.06$ (Equation 7.1).

Grip detection. Knowing the grip allows CSAW to choose the appropriate correlator module for correlation and give a reliable confidence score; if the phone is not in the wristband-hand, then correlation score from M2MC is meaningless. We evaluate detection accuracies for the three grips: BB, WN, and NW (Figure 7.2). The orientation of the phone’s

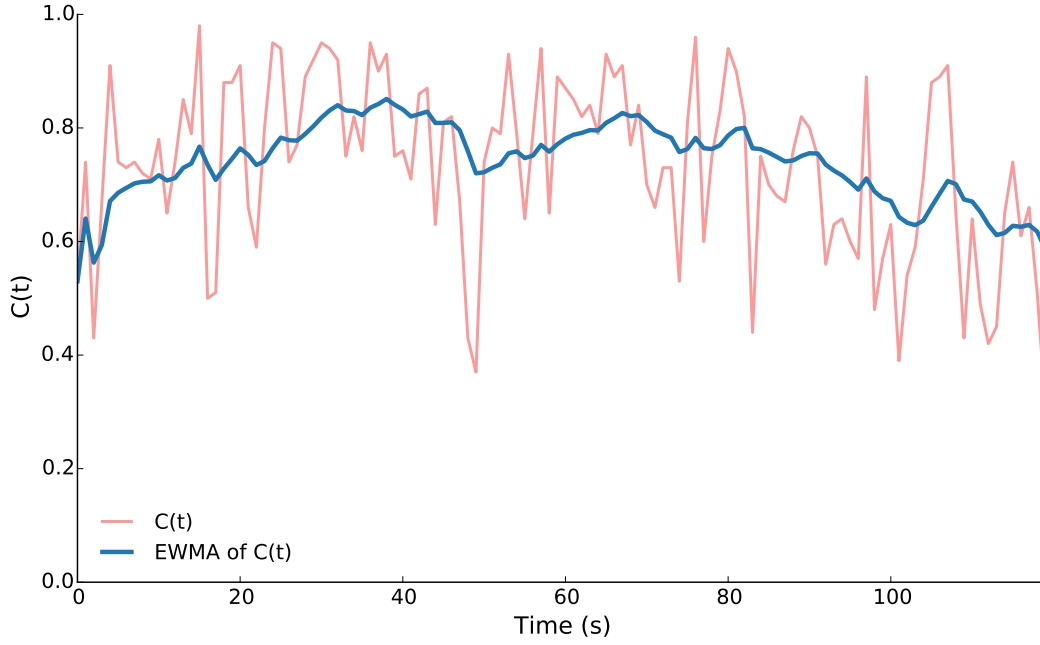


Figure 7.3: Instantaneous confidence score for participant P1.

Table 7.3: Average false-positive rate (FPR) and balanced accuracy (BAC) across all participants for continuous authentication, with $w = 4$ s and $\alpha_m = 0.5$; mean \pm standard-deviation.

Activity	FPR	BAC
Stationary	0.016 ± 0.001	0.980 ± 0.004
Walking	0.017 ± 0.001	0.979 ± 0.004
PC use	0.017 ± 0.001	0.980 ± 0.004
Phone	0.049 ± 0.003	0.964 ± 0.004
Same	0.072 ± 0.003	0.952 ± 0.005
All cases	0.019 ± 0.001	0.978 ± 0.004

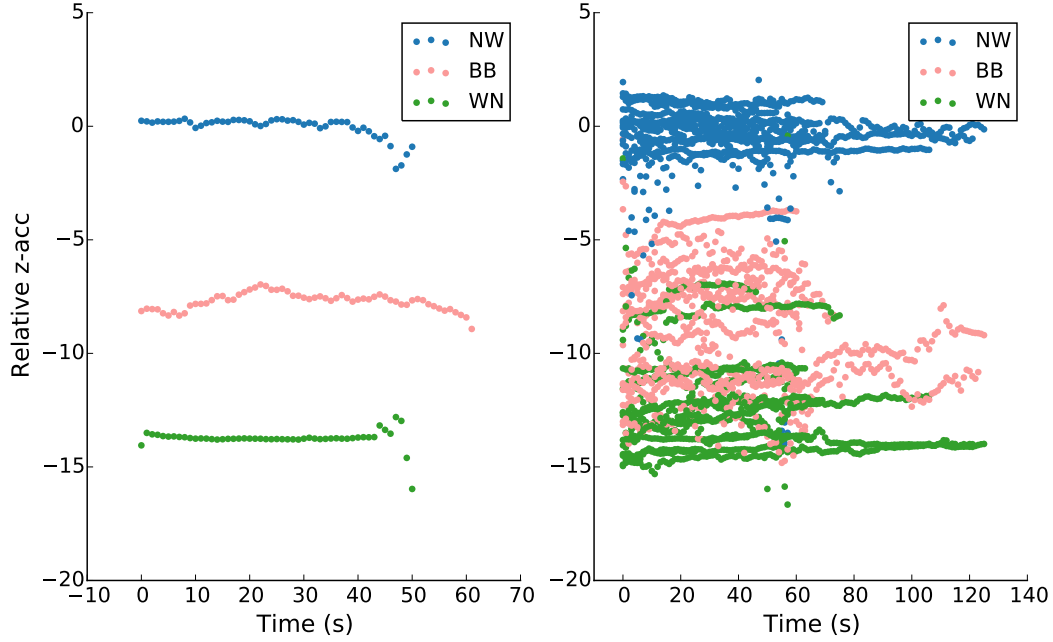


Figure 7.4: Wristband acceleration (along z -axis) relative to the phone for a) participant P1, and b) all participants, over time.

touchscreen and the wristband’s face can be determined by the z -axis component of their acceleration. To measure the wristband’s relative face orientation with respect to the phone’s touchscreen, we compute the difference of their z -axis acceleration. Figure 7.4a shows a scatter plot of the z -axis acceleration of the wristband relative to the phone for participant S1, and Figure 7.4b shows the same plot for all participants. We can distinguish these three grips for S1 with two simple thresholds, but choosing a generic threshold that applies to everyone is tricky, as Figure 7.4b shows. For CSAW it is critical to distinguish between grips NW and WN, as it uses these grips to choose the correlation module. Using a threshold of -5, we can distinguish between NW and WN grips in our dataset with 99.12% accuracy. Although in our user study participants performed different tasks using different apps, a larger user study will verify how well this threshold serves as a generic threshold. (Another approach is to build a classifier to distinguish cases.)

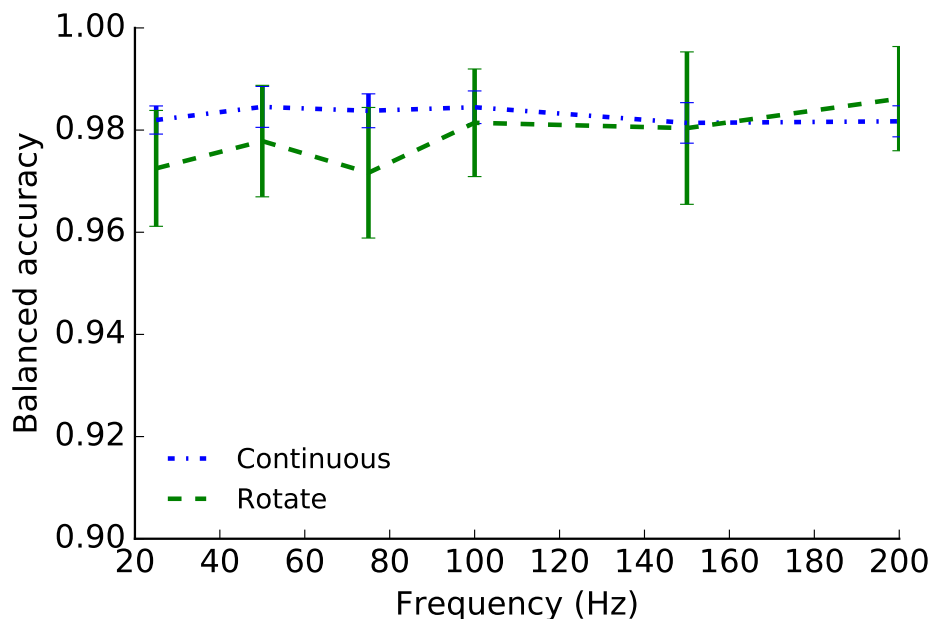


Figure 7.5: Effect of sampling rate on balanced accuracy (BAC) for continuous authentication and the rotate maneuver.

Sampling rate and window size. CSAW depends on a steady stream of observations of phone and wrist motion; except where noted, our experiments used a 200 Hz sampling rate for the accelerometer and gyroscope on both the phone and wristband. How does the sampling rate affect accuracy of the confidence score, or ultimately, the authentication decisions? A lower sampling rate would allow use of cheaper hardware and reduces power and bandwidth consumption on both the wristband and the phone; how low can we go and maintain accuracy? Figure 7.5 shows the BAC for continuous authentication and for initial authentication with the rotate action, for different sampling rates. The negative test cases for the continuous authentication are generated for the scenario where the adversary is using the phone and the user is walking, stationary, or using a PC; the negative test cases for initial authentication with the rotate action are generated for the scenario where the user performs the rotate action on another phone or object and the adversary performs rotates the phone at the same time, but the adversary does not mimic the user.

Many participants in our study performed the rotate action very quickly, and the increase in the BAC for the rotate action suggests that a higher sampling rate helped in capturing and correlating the quick rotate action. Continuous authentication BAC was largely unaffected by the sampling rate. We suspect that, during phone use, the wrist and phone motions are low-frequency motions, which can be captured sufficiently with low sampling rates. Overall, these results imply that a 40 Hz sample rate is sufficient during continuous authentication, but a higher 200 Hz rate may be useful when the user is asked to conduct the rotate maneuver.

In Section 7.2 we noted that CSAW’s correlation modules use a window of motion data for their determinations; except where noted, all of our experiments used 1 s window when the phone was locked (initial authentication cases) and 2 s window when the phone was unlocked (continuous authentication cases). Would larger windows provide better accuracy? Larger windows could be especially helpful for continuous authentication, in which CSAW produces a steady series of regular confidence scores but which could easily leverage a larger window of data in producing each score. Figure 7.6 shows the BAC for continuous authentication for the ‘phone’ use case in Table 7.3 for window sizes from 2 sec to 10 sec, sample rate of 200 Hz, and overlap fraction of 0.5. The values in the plot are averages over 10-fold cross-validation. We believe the accuracy increases for larger window sizes because there is more motion that CSAW can leverage to correlate and boost its confidence score; however, a larger window size reduces its responsiveness to change and may delay its determination that an imposter has taken over the phone.

Sensors and Features. Recall from Section 7.2 that M2MC uses 256 features for motion-to-motion correlation, including 40 temporal features. We hypothesized that temporal features are important in motion correlation to achieve low FNR and FPR required for

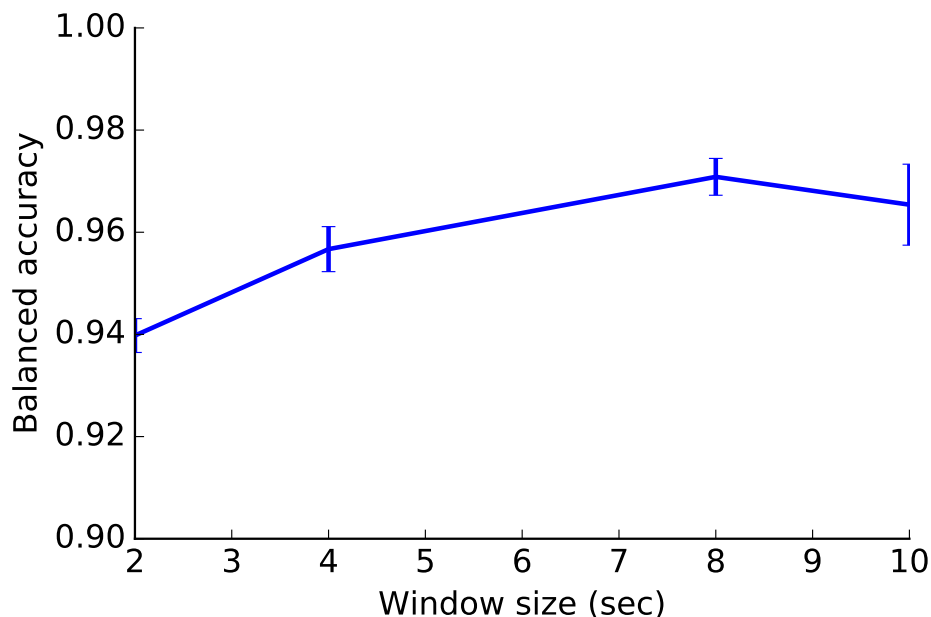


Figure 7.6: Effect of window size on balanced accuracy (BAC) for continuous authentication.

secure applications. Consider the ‘phone’ use case in Table 7.3 with 200 Hz sampling rate, window size of 4 s, and overlap of 0.5; Table 7.4 shows that the temporal features clearly had a substantial impact on accuracy: the first and last rows show CSAW accuracy without and with those temporal features. Table 7.4 also shows the accuracy when using only magnitudes (second row) versus magnitudes plus all three axes’ data (bottom row); the use of the individual axes made a substantial difference. Table 7.4’s bottom three rows show the accuracy for phone use if CSAW uses only the accelerometer or only gyroscope sensors, or when both sensors are used. Accuracy was substantially improved if both sensors’ data were involved.

Summary. In short, CSAW’s confidence scores proved to be remarkably accurate when used for initial and continuous authentication, even when used by a relatively naive threshold policy. These results could be achieved with a moderate sampling rate (40 Hz) in most cases, benefited by including data from both accelerometer and gyroscope, from all three axes, and

Table 7.4: Average false-positive rate (FPR), false-negative rate (FNR), and balanced accuracy (BAC) during continuous phone use for different subsets of features and sensors; mean \pm standard-deviation.

Features	FPR	FNR	BAC
No temporal	0.074 ± 0.011	0.040 ± 0.008	0.943 ± 0.006
Only magnitudes	0.062 ± 0.009	0.044 ± 0.006	0.947 ± 0.005
Only accelerometer	0.081 ± 0.010	0.023 ± 0.006	0.948 ± 0.003
Only gyroscope	0.091 ± 0.015	0.055 ± 0.009	0.927 ± 0.007
All	0.048 ± 0.007	0.021 ± 0.007	0.965 ± 0.005

leveraging temporal features. CSAW confidence scores should be a rich and reliable source of information on which to build more sophisticated authentication-related policy decisions.

7.3.3 Security

In this section, we evaluate the quality of CSAW to serve as a secure foundation for phone authentication. First, though, we must consider our adversary. Recall, from Section 2.3, our adversary is a malicious individual with physical access to the phone, a curious family member or friend, or a curious colleague. The adversary may even be another authorized user (e.g., if the phone is a shared device and each authorized user is allowed access to only certain apps or information). Our adversary seeks to achieve one of two related goals: *opportunistic snooping*, the adversary takes the opportunity to snoop around the owner’s phone when the owner is not near her phone; and *stealing credentials*, the adversary steals the owner’s credentials (e.g., passcode) so he can use them to access the owner’s device in her absence or attack her accounts on other devices or websites. In the face of the above threats, we can now describe how CSAW offers the following security benefits.

Verify the owner: CSAW should verify the individual who is actually using the phone. CSAW continuously provides a quantitative score regarding its confidence that the owner is

the person actually using the device, based on 1) the proximity of that the owner's wristband (in radio range and able to transmit wrist-movement data to the phone) and 2) the correlation of the owner's wrist movements with the phone's observed motion and inputs. Unlike approaches based purely on physical proximity, it is not sufficient for the owner to be near the device. In Section 7.3.2, we show that CSAW was highly accurate in determining whether the user was indeed the owner, that is, with fewer than 2% false-positive results.

Continuous: CSAW should continuously authenticate the owner when she is interacting with her phone, and deauthenticate (lock) when anyone else tries to use her phone. CSAW is specifically designed to continuously provide a quantitative score regarding its confidence that the owner is the person actually using the device, enabling the phone's operating system to deauthenticate the user whenever the confidence dips below a certain threshold.

Require intent: CSAW should authenticate a user to her phone only when she intends to use her phone. Although CSAW could be configured to turn on the screen and unlock the phone whenever a pick-up gesture is detected, it is unlikely that such behavior would be desired. For initial authentication, therefore, we anticipate CSAW to be configured to require some expression of user intent, such as turning on the display of the phone.

Resilient to physical observation: An attacker cannot impersonate the user after observing her authenticate one or more times. Attacks include shoulder surfing, filming [83, 102], or thermal imaging [61]. Since CSAW does not require the user to enter an authentication secret, none of these attack methods are available to the adversary.

7.3.4 Usability

Results from our user study that show that the false-negative rates for CSAW were less than 2%. In other words, the fraction of the time that CSAW reported a low score when the legitimate user was indeed using the phone was small. CSAW offers following usability benefits.

Quick: CSAW can leverage user’s natural pick-up action for initial authentication, without requiring the user to do any explicit action, hence it is quick. However, when the initial authentication fails, to boost the confidence score for initial authentication, CSAW may ask the user to do the rotate maneuver, which is easy and quick to perform; our study participants took on 1.492 ± 0.351 seconds. CSAW is also quick to deauthenticate – it provides a confidence score every two seconds and when there is not user-phone interaction the confidence score drops to zero quickly.

Effortless: CSAW does not require the user to remember any secret for authentication. Because CSAW leverages the user’s natural interactions using her phone, such as picking it up, holding the phone in the hand providing input, authentication in CSAW is almost effortless. For authentication with pick-up action the user need not perform any explicit task; when the initial authentication through the pick-up action fails, the user needs to perform the rotate maneuver, which is easy and quick to perform.

7.4 Discussion

CSAW works remarkably well, but there are some limitations.

Energy efficiency: CSAW depends on continuous (or frequent) readings from accelerometer and gyroscope sensors in both the phone and wristband, and a continuous feed of these readings from the wristband to the phone over BLE. Such activities will affect battery lifetime in these battery-limited devices. In a production system the phone would leverage a low-power coprocessor (common in most phones today) to monitor phone motion and minimize the work imposed on the main processor; it could also inform the wristband about when readings are needed so the wristband can sleep when the phone is not in use or a pick-up gesture is unlikely. In our prototype we did not have access to the phone's motion coprocessor so we were unable to measure these energy demands in a realistic setting.

Single-handed use: CSAW performs well when the wristband is worn on the hand that holds the phone or on the hand that touches the phone screen. There are moments, however, when the owner's wristband hand may be uninvolved – neither holding the phone nor touching the phone; for example, the user may wear the wristband on her right hand but use only her left hand to hold the phone and touch the screen. In such cases, CSAW is unable to verify the owner as the user, and the phone would have to rely on alternative means for authentication (e.g. to require a PIN code). In practice, we believe that the gains in usability outweigh the inconvenience of minor deviations from natural interactions caused by this issue. Also, if CSAW is used as a second-factor authentication mechanism, then the security gains also outweigh this issue.

Bluetooth pairing: Some use cases (particularly access limitation and delegation) require the wireless technology to allow pairing multiple wristbands to multiple phones, and allow simultaneous communications between a wristband and multiple phones. Bluetooth Low Energy (BLE) 4.1 supports scatternet operation so this capability is now becoming available.

Such cases are more likely for tablets than smartphones; we envision a user working with multiple phones and tablets, or a school/workplace in which tablets are shared among several users.

7.5 Summary

People’s different phone usage behavior and different security expectations require an authentication scheme that can adapt to their needs. CSAW provides a continuous user verification score, quantitative estimate of the confidence that the individual using phone is in fact the phone’s owner, that enables applications and phone OS to implement flexible authentication policies that meet the user’s security expectations. When the phone is in use, CSAW continuously monitors the phone owner’s wrist movement and correlates it with the phone’s motion and inputs to determine whether the phone is in the owner’s hand. CSAW also provides a quick initial authentication leveraging the natural phone pick-up action when performed with the wristband hand.

CSAW succeeds because it incorporates the following novel technical approaches: (i) a hybrid correlation model that: (a) recognizes how the owner is holding the phone and (b) once a grip is classified, correlates either phone motion or phone input with wrist motion; (ii) leverages three-axis motion features about phone and wristband orientation; and (iii) includes temporal features that require any adversary to synchronously mimic the owner in order to use or unlock the owner’s phone.

Our evaluation from an in-lab user study shows that CSAW can be used to authenticate users with 99% accuracy using the simple natural phone pick-up action. During continuous phone use, CSAW could verify the user with 96.5% accuracy every 2 s. CSAW’s confidence score can feed a variety of authentication-decision algorithms, allowing each to be tuned to

achieve high security (very low FPR) or high usability (low FNR).

8

Comparative evaluation

We compare CSAW with other related authentication schemes using the Usability Deployability Security (UDS) evaluation framework [13]. The UDS framework defines a set of properties to evaluate web authentication schemes, but many of those properties are relevant to computer authentication schemes. We first define the properties we use for evaluation and then present our evaluation.

8.1 Benefits

The UDS framework defines 25 properties to evaluate web authentication schemes: 14 usability and deployability properties and 11 security properties. Out of those 25 properties, we chose the 12 properties that are relevant to evaluating an authentication scheme for a computer (instead of a web service or website). We use an additional security property *Continuous-User-Verification* in our evaluation. As in the UDS framework, we rate each scheme as either offering or not offering the benefit of a property; if a scheme *almost* offers the benefit, but not quite, we indicate this with the *Quasi-* prefix.

8.1.1 Usability benefits

- U1 *Memorywise-Effortless*: Users of the scheme do not have to remember any secrets at all.
- U2 *Nothing-to-Carry*: Users do not need to carry an additional physical object to use the scheme. We grant schemes *Quasi-Nothing-to-Carry* benefit if the scheme can be implemented on an object that users carry or wear anyway, such as their mobile phone, wrist watch, wearable fitness devices.
- U3 *Physically-Effortless*: The authentication process does not require physical user effort beyond, say, pressing a button. Typing, scribbling, performing a set of motions counts as physical effort. We grant schemes the *Quasi-Physically-Effortless* benefit if the user's effort is limited to natural actions such as speaking. We consider tapping a key five times as *Quasi-Physically-Effortless* because it is easier than typing a password, and tapping a key, any key, is more natural to people than typing a specific password.
- U4 *Easy-to-Learn*: Users who do not know the scheme can figure it out and learn it without too much trouble, and then easily recall how to use it.

- U5 *Efficient-to-Use*: The time the user must spend for each authentication attempt is acceptably short. It is acceptable if the scheme requires more time for a one-time setup, e.g., providing multiple fingerprint scans when setting up fingerprint authentication, pairing a token with the computer.
- U6 *Easy-Recovery-from-Loss*: A user can conveniently regain the ability to authenticate if the token is lost or the credentials are forgotten.
- U7 *Accessible*: Users who can use CSAW are not prevented from using the scheme by disabilities or other physical (not cognitive) conditions. CSAW does not require visual attention to perform initial authentication, so it is accessible to visually impaired users; hence, schemes that require visual attention are rated as *Quasi-Accessible*.

8.1.2 Security benefits

- S1 *Resilient-to-Physical-Observation*: An attacker cannot impersonate a user after observing them authenticate one or more times. Attacks include shoulder surfing, filming the keyboard or mouse use [73], recording keystroke timings based on sensors near the keyboard [53], or thermal imaging the keypad [61].
- S2 *Resilient-to-Leaks-from-Other-Verifiers*: Any information gained by an attacker from one computer cannot be used to impersonate the user to another computer.
- S3 *Resilient-to-Phishing*: An attacker who simulates the authentication process, e.g., by spoofing the authentication screen, cannot collect credentials that can later be used to impersonate the user on the actual computer.
- S4 *Resilient-to-Theft*: If the credentials are lost they cannot be used for authentication by another person who gains possession of it. The lost credentials can be passwords written down by paper or hardware tokens.

This definition is slightly different than the UDS definition. In the UDS framework, this benefit is considered only for schemes in which physical objects are used for authentication; we consider theft of even non-physical credentials such as passwords (which can be stolen when people write them down) or fingerprints (which can be lifted from surfaces). As in the UDS framework, we grant *Quasi-Resilient-to-Theft* if the scheme protects the credential with the modest strength of a PIN.

- S5 *Require-Intentionality*: The user is authenticated only with the user’s consent or intent. The UDS benefit requires an ‘explicit’ consent, which can make a continuous authentication scheme unusable, so we remove the ‘explicit’ requirement, and allow schemes to use a passive consent or implicitly recognize the user’s intent.
- S6 *Continuous-User-Verification*: The authentication method should continuously verify the user while she is using the computer, without any effort from the user.

8.2 Evaluation of authentication schemes

We evaluate authentication schemes that are based on passwords, proximity, fingerprint, voice, face, keystroke dynamics, and touchscreen dynamics. We also evaluate a token-based authentication scheme (called PICO), two proximity-based authentication schemes, and a biometric authentication scheme based on impedance [74].

8.2.1 Passwords

Passwords clearly are not *Memorywise-Effortless*. They provide *Nothing-to-Carry* benefit, but they are not *Physically-Effortless* as passwords need to be typed. Passwords are *Easy-to-Learn* and *Efficient-to-Use*, and they offer *Easy-Recovery-from-Loss* as they can be easily reset. We rate them *Quasi-Accessible* because they require visual attention; visually impaired

users struggle with passwords as Dosono et al. discovered in their study [23].

Passwords are not *Resilient-to-Physical-Observation* as they can be easily captured by shoulder surfing [93] or by filming [8], and for the same reason (i.e., they can be easily stolen) they are also not *Resilient-to-Theft*. They are not *Resilient-to-Leaks-from-Other-Verifiers* if the user has same password for different computers. They are also not *Resilient-to-Phishing* as the attacker can use the obtained password to impersonate the user. They do *Require-Intentionality* as users have to enter their passwords to authenticate. Passwords are one-time authentication schemes and do not offer *Continuous-User-Verification*.

8.2.2 PICO

Stajano proposed a token-based authentication scheme called PICO as a replacement for passwords [90]. In PICO, the user carries a small device (a token) that has two buttons, a small display, a camera, and a radio to communicate to the computer. During authentication, the computer displays a 2D visual code encoded with credentials required for authentication. The user takes a picture of the 2D code with the camera on the token, which then processes the code and sends the credentials to the computer completing the authentication process. PICO was designed to be and is *Memorywise-Effortless*. We rate it *Quasi-Physically-Effortless* and *Quasi-Efficient-to-Use* because the user has to align the camera to take a picture of the code displayed on the computer. It is not *Easy-to-Learn* because in addition to learning to find and take photo during authentication, the user has to manage PICO siblings (additional mobile devices that the user carries) with the purpose to avoid misuse if the token is lost. We grant it *Quasi-Easy-Recovery-from-Loss* because token or the siblings need to be replaced if they are lost. Although PICO requires visual attention, it does not require typing passwords, so we rate it as *Accessible*, but worse than CSAW.

Because of the secure communication between the token and computer, PICO is *Resilient-to-Physical-Observation* and *Resilient-to-Phishing*. It is *Resilient-to-Leaks-from-Other-Verifiers* because it uses different credentials for different computers. We rate it *Quasi-Resilient-to-Theft* assuming the token is secured by PICO siblings. PICO does *Require-Intentionality* and we grant it *Continuous-User-Verification*, but rate it worse than CSAW because it uses proximity for continuous authentication.

8.2.3 Proximity

We discuss two proximity-based authentication schemes: one that requires a hardware token and software changes to the computer (ZIA) and another that does not require any software change on the computer (hardware-based).

ZIA. Zero-Interaction Authentication (ZIA) is a proximity-based authentication scheme proposed by Corner and Noble [18]. In ZIA, the user carries a wireless authentication token and while the user is in the radio range, the computer stays authenticated to the user. ZIA is *Memorywise-Effortless* and *Physically-Effortless* – indeed, it is zero-effort; it is *Easy-to-Learn* and *Efficient-to-Use*. We rate ZIA as *Quasi-Nothing-to-Carry* because the proximity token can be integrated with their smartphones. Tokens are not as easy to recover as passwords, but they can be recovered by buying another token, so we grant ZIA *Quasi-Easy-Recovery-from-Loss*. ZIA is *Accessible* and we rate it better than CSAW.

ZIA is *Resilient-to-Physical-Observation*, *Resilient-to-Leaks-from-Other-Verifiers*, and *Resilient-to-Phishing* if we assume that the communication between the token and the device is secure and cannot be eavesdropped by the attacker. We grant it *Quasi-Resilient-to-Theft* because the tokens can be secured with a PIN. ZIA does not *Require-Intentionality* as the user is authenticated whenever she is in the proximity of the device, without any consent. We

grant it *Continuous-User-Verification*, but rate it worse than CSAW because it uses proximity for continuous authentication.

Hardware-based. Landwehr proposed this scheme to protect unattended computers without modifying the software on the computer – a pure hardware-based solution [43]. In this scheme, the computer is instrumented with a detector device through which the keyboard and the mouse are connected to the computer; thus, the detector device can connect or disconnect the keyboard and the mouse. Users carry a token and when they are near the computer and associated with the detector, the detector connects the keyboard and the mouse, otherwise keeps them disconnected. This scheme is *Memorywise-Effortless* and *Physically-Effortless*. It is *Easy-to-Learn* and *Efficient-to-Use* as users do not have to do much except walk to a computer terminal and use them. This scheme is *Quasi-Nothing-to-Carry* as the proximity token can be integrated with the user's smartphone, and because of this scheme requires token it is *Quasi-Easy-Recovery-from-Loss*. This scheme is *Accessible* and better than CSAW, because it is zero-effort for the purpose of authentication, and hence, anyone who can use a computer can use this scheme.

We rate this scheme *Resilient-to-Physical-Observation*, *Resilient-to-Leaks-from-Other-Verifiers*, and *Resilient-to-Phishing* if we assume secure communication between the token and the detector device. Token can be secured with a PIN, so we grant it *Quasi-Resilient-to-Theft*. We grant this scheme *Quasi-Require-Intentionality* because when two authorized users are near the target computer, the scheme involves the user in the authentication process to identify the right user. This scheme does offer *Continuous-User-Verification*, but rate it worse than CSAW because of its reliance on proximity for continuous authentication.

8.2.4 Fingerprint recognition

Fingerprint-based schemes are *Memorywise-Effortless* and *Nothing-to-Carry*, as in any biometric scheme, but they are not *Physically-Effortless* as the user has to swipe or hold the finger on the reader. They are *Easy-to-Learn* and *Efficient-to-Use*, but they do not offer *Easy-Recovery-from-Loss* as a fingerprint, once stolen, cannot be reset. Fingerprints are not effective against users who do not have fingerprints or if the user's fingerprint changes due to external factors, such as an injury; indeed, we noticed a high failure rate with fingerprints in our user study (Section 4.7.3). Furthermore, visually impaired people sometimes find it difficult to align their finger to the fingerprint reader [23]. So, we rate them *Quasi-Accessible*.

They are *Resilient-to-Physical-Observation* as they are hard to capture by observing the user or by filming. These schemes are not *Resilient-to-Leaks-from-Other-Verifiers*, *Resilient-to-Phishing*, and *Resilient-to-Theft*. Users have to swipe their finger to get authenticated so they do *Require-Intentionality*, but they do not offer *Continuous-User-Verification*.

8.2.5 Voice recognition

Voice-based authentication schemes are physiological biometric schemes that authenticate users based on their unique voice patterns. Voice-based authentication can be used for initial authentication, as for passwords or fingerprints, with many of the same properties as fingerprints. However, voice-based methods can also be used for continuous authentication, which is how we evaluate them here.

These schemes are also *Memorywise-Effortless* and *Nothing-to-Carry* but they are not *Physically-Effortless* as the user has to speak out loud for authentication. For the purpose of initial authentication, we would grant them *Quasi-Physically-Effortless*, but for the purpose of continuous authentication, the user would have to continue speaking, so we do not grant

them the *Physically-Effortless* benefit. They are *Easy-to-Learn* and *Efficient-to-Use*. They do not offer *Easy-Recovery-from-Loss*, like other biometrics. We grant them *Quasi-Accessible* because they do not work for speech-impaired users.

They are not *Resilient-to-Physical-Observation* as an adversary can easily record a user's voice with a microphone. These schemes are also not *Resilient-to-Leaks-from-Other-Verifiers* and *Resilient-to-Phishing*, and once the voice credentials are stolen an adversary can use them to impersonate the user so these schemes are not *Resilient-to-Theft*. Users have speak out loud to authenticate, so we grant these schemes *Quasi-Require-Intentionality*, because without some hot-word, it is difficult to distinguish between a user speaking to a colleague or expressing intent to authenticate to the computer. These schemes can be used to perform *Continuous-User-Verification*.

8.2.6 Face recognition

Face-based schemes authenticate a user based on the computer's recognition of the image of her face as captured by the computer's camera. Face-based authentication can be used for initial authentication, as fingerprints, with many of the same properties as passwords. Face-based methods can also be used for continuous authentication, which is how we evaluate them here.

These schemes are *Memorywise-Effortless* and *Nothing-to-Carry*, like any biometric, and these schemes are also *Physically-Effortless* as the user simply has to be in-front of the camera when using the device. These schemes, like voice and fingerprint biometrics, do not provide *Easy-Recovery-from-Loss*. These schemes are *Easy-to-Learn*, but we grant them *Quasi-Efficient-to-Use* because of the user steps involved to face the camera and stay in the camera's view while using the computer (for continuous authentication). We grant them

Quasi-Accessible because these schemes may not perform with wearables (e.g., glasses), in poor lighting, or if the user is wearing a mask (e.g., surgical mask [86]).

These schemes are not *Resilient-to-Physical-Observation* as an adversary can easily capture an image of the user's face with a camera. These schemes are also not *Resilient-to-Leaks-from-Other-Verifiers*, *Resilient-to-Phishing*, and *Resilient-to-Theft*. We grant these schemes *Quasi-Require-Intentionality* because they require the user to be within the camera's field of vision, which is a better indication of intent to use a device than simply being in proximity of the device. These schemes can provide *Continuous-User-Verification*.

8.2.7 Impedance

These schemes authenticate a user based on her body's impedance response to small electric current passed through her body [17, 74]. We rate impedance-based authentication schemes based on the continuous authentication scheme proposed by Rasmussen et al., in which the keyboard of the computer is instrumented to measure the impedance response of the user's body [74].

Like other biometric scheme, these schemes are *Memorywise-Effortless* and *Nothing-to-Carry*. We consider these schemes *Physically-Effortless* because these schemes can authenticate when the user is using the device (i.e., typing) without requiring the user to provide any explicit input. These schemes do not provide *Easy-Recovery-from-Loss*, but they are *Easy-to-Learn* and *Efficient-to-Use*. We grant them *Quasi-Accessible* because human body's impedance response can change due to external environmental factors such as temperature or water intake of the user.

These schemes are *Resilient-to-Physical-Observation* but they are not *Resilient-to-Phishing*, which could be used to steal the user's unique impedance response. They

schemes can be made *Resilient-to-Leaks-from-Other-Verifiers* by using a different frequency-dependent response for each device so that a credential leaked from a device cannot be used to authenticate on another device. Although an adversary can use the stolen credentials to impersonate the user on the device, these schemes can prevent the adversary from using the credentials for one device on another, so we grant these schemes *Quasi-Resilient-to-Theft*. As typing indicates intent to use a device, these schemes *Require-Intentionality* and they offer *Continuous-User-Verification*.

8.2.8 Keystroke dynamics

Keystroke-based schemes are typically used for initial authentication, we rate them as such. In keystroke-based initial authenticate schemes users type a password (either secret or known) and they are authenticated based on their unique typing behavior. These schemes are *Memorywise-Effortless* and users have *Nothing-to-Carry*. They are not *Physically-Effortless* because the user has to type a password. These schemes, like some other biometric schemes, do not provide *Easy-Recovery-from-Loss* and we grant them *Quasi-Accessible*, similar to passwords. These schemes are *Easy-to-Learn* as the user does not have to learn anything new (other than typing password), and they are *Efficient-to-Use*.

These schemes are not *Resilient-to-Physical-Observation* as the keystroke dynamics can be recorded using a sensor near the keyboard [53]. They are also not *Resilient-to-Leaks-from-Other-Verifiers* and *Resilient-to-Phishing*. These schemes are not *Resilient-to-Theft* because, once the keystroke dynamics of a user is stolen, an adversary can use them to impersonate the user [55]. As the user has to type in order to be authenticated and typing indicates the intent to use a device, these schemes *Require-Intentionality* to authenticate. We rated these schemes as initial authentication schemes, so we do not grant them *Continuous-User-Verification*.

8.2.9 Touch input dynamics

In these authentication schemes, the user is authenticated based on her unique pattern of providing inputs on a touchscreen. Example schemes include the re-authentication system proposed by Li et al. [45] and the SilentSense system proposed by Bo et al. [12]. These schemes are used are developed for continuous authentication and we rate them as such.

These schemes are *Memorywise-Effortless* and *Physically-Effortless*, and they offer *Nothing-to-Carry* benefit. These schemes, like some other biometric schemes, do not provide *Easy-Recovery-from-Loss* and we grant them *Accessible* because anyone who can use a smartphone can use this scheme. Users do not have to explicitly perform any action for authentication, so these schemes are *Easy-to-Learn* and they are *Efficient-to-Use*.

These schemes are *Resilient-to-Physical-Observation* as it is difficult to capture touch behavior by observing the user. They are, however, not *Resilient-to-Leaks-from-Other-Verifiers* and *Resilient-to-Phishing*, as an adversary can learn user's behavior from the input. These schemes are not *Resilient-to-Theft* because, once the touch behavior of a user is stolen an adversary can use it to impersonate the user; Serwadda and Phoha demonstrated an attack where a robot provides touch inputs to a smartphone based on users' learned behavior [80]. Touch inputs from a user implies her intends to use the phone, so we rate them *Require-Intentionality* and they do offer *Continuous-User-Verification*.

8.3 Conclusion

Table 8.1 summarizes our evaluation of CSAW and the ten other authentication schemes, providing a concise overview of the benefits that different authentication schemes offer. A perfect authentication scheme would offer all the benefits (filled circles in all the columns

in the table), but among the schemes we considered, no scheme is perfect. Biometric schemes offer the usability benefit of not having to carry anything over CSAW, but they have serious security concerns: once stolen, biometrics cannot be reset, and they are not difficult to steal; CSAW rates much better compared to biometric solutions in security. Proximity-based authentication schemes, however, come close to the perfect scheme: they offer ten out of thirteen benefits. Proximity-base schemes, like any token-based scheme, suffer from the drawback that the user has to carry a token and it is not always easy to recover (or reset) a lost token. However, tokens can be integrated with smartphones and wearables, which are becoming ubiquitous, so these drawbacks are perhaps not as severe usability concerns as they were few years ago. Our concern with proximity-based schemes is, however, related to their security: they do not require user intentionality for authentication and their continuous authentication approach, which is based on distance, may leave the computer vulnerable when the user steps away. CSAW builds on proximity-based solutions, offering the same usability benefits, but also offering the missing security benefits. CSAW offers user intentionality at a small usability trade-off (the user must perform a simple action to express her intent to authenticate) and it offers a secure continuous authentication wherein user is authenticated only while she is using – providing inputs to – the computer.

We want to emphasize that the rating we granted to various authentication schemes are through our understanding and analysis of the schemes. We evaluated the schemes objectively and it is reassuring that most of our ratings match with the ratings given by others (Bonneau et al. [13]). However, our evaluation of CSAW is based only on studies conducted in laboratory and our analysis of the system; when CSAW is evaluated in uncontrolled settings its ratings may change.

Table 8.1: Comparative evaluation of CSAW and other authentication schemes.

Scheme	References	Usability							Security					
		<i>Memorywise-Effortless</i>	<i>Nothing-to-Carry</i>	<i>Physically-Effortless</i>	<i>Easy-to-Learn</i>	<i>Efficient-to-Use</i>	<i>Easy-Recovery-from-Loss</i>	<i>Accessible</i>	<i>Resilient-to-Physical-Observation</i>	<i>Resilient-to-Leaks-from-Other-Verifiers</i>	<i>Resilient-to-Phishing</i>	<i>Resilient-to-Theft</i>	<i>Require-Intentionality</i>	<i>Continuous-User-Verification</i>
CSAW		●	○	○	●	●	○	●	●	●	●	○	●	●
Passwords			●		●	●	●	○					●	
PICO	[90]	●	○	○		○	○	●	●	●	●	○	●	○
Proximity-based														
ZIA	[19]	●	○	●	●	●	○	●	●	●	●	○		○
Hardware-based	[43]	●	○	●	●	●	○	●	●	●	●	○	○	○
Biometric														
Fingerprint	[77]	●	●	○	●	●		○	●				●	
Voice	[3]	●	●		●	●		○					○	●
Face	[5]	●	●	●	○	○		○					○	●
Impedance	[74]	●	●	●	●	●		●	●	○		○	●	●
Keystroke	[22]	●	●		●	●		○					●	
Touch input	[45]	●	●	●	●	●		●	●				●	●

● = offers the benefit; ○ = almost offers the benefit; *no circle* = does not offer the benefit.
 ||| = better than CSAW; || = worse than CSAW; *no pattern* = equivalent to CSAW.

9

Summary and Future work

In this dissertation we explored problems associated with user authentication, and developed an authentication scheme to alleviate some of these problems. We conducted a digital diary study with twenty six participants to understand their authentication behavior; participants logged when, how, and where they performed authentications for one week, and then we conducted semi-structured interviews with them to gain a deeper understanding of their authentication habits, opinions, and frustrations. In our study, we observed that the number

of authentications to physical objects (e.g., cars, doors) was significant (about 22%), but a majority of the authentications were to phone, laptop, and websites (in that order), and about 75% of authentications were performed with passwords (or PIN). We found that for password-based authentications, participants encountered most failures with websites (about 7%), followed with laptops and desktops (about 5%), and least with phones (about 2%). We observed that participants' authentication patterns were fairly consistent throughout a week; we found some variance in authentication behavior across age, but no variance across gender. Overall, we found that authentication was a noticeable part of all participants' lives and burdensome for many participants, but all accepted it as the cost of security, devising their own ways to cope with the burden.

We proposed an authentication scheme, called Continuous Seamless Authentication using Wristbands (CSAW), for desktops and smartphones. CSAW provides initial authentication, continuous authentication, and automatic deauthentication. We designed CSAW to be unobtrusive and secure with an initial authentication process that blends into peoples' workflow. In CSAW, users wear a wristband and they authenticate to a desktop computer by simply tapping a key five times; while they use the desktop CSAW continues to verify their presence by correlating their wrist movement with their typing on the keyboard or their use of the mouse. When a user stops using the desktop or steps away, CSAW can deauthenticate the user; actually, CSAW offers four different parameters to express when to deauthenticate a user, enabling flexible deauthentication policies.

In the case of smartphones, when a CSAW user picks up her phone with her wristband hand, she gets authenticated by the end of her pick-up action – CSAW correlates the pick-up motion of the phone and the user's wristband to authenticate the user. During phone use, CSAW provides a continuous user verification score, a quantitative estimate of the

confidence that the individual using the phone is in fact the phone's owner, that enables applications and phone OS to implement flexible authentication policies that meet users' security expectations.

We conducted several laboratory user studies to evaluate CSAW's performance from the system's perspective as well as the user's perspective. In CSAW, initial authentication is efficient (1-2 seconds) and participants found it to be effortless and unobtrusive. For continuous authentication on desktops, CSAW had an average FNR and FPR of 0.037 (\pm 0.096) and 0.031 (\pm 0.057), respectively; in our user study, CSAW was able to identify all adversaries within 50 seconds. For smartphones, CSAW was able to authenticate users with 99% accuracy using the simple natural phone pick-up action, and during continuous phone use, CSAW could verify the user with 96.5% accuracy every 2 seconds.

Based on the evaluation, CSAW shows promise for reducing users' authentication burden for desktops and smartphones. However, to use CSAW in the real world there is more work to be done; we discuss this future work below.

Improvement to continuous authentication for desktops

CSAW has two limitations with its continuous authentication approach for desktops: 1) the wristband must be worn the user's mouse-hand, and 2) the user must use both the keyboard and the mouse, otherwise CSAW cannot verify the user (e.g., if she is mostly using keyboard) with high confidence. These limitations can be eliminated if CSAW can verify the user using only her typing interactions: most people type with both hands, so they can wear the wristband on either hand. While typing, the small wrist movements could be correlated to keystrokes to verify the user. Recent work showed that in controlled settings certain keystrokes can be inferred using the motion sensor data from a user's smartwatch [48,

99]. For the purpose of authentication, we would have to find strong correlation between keystrokes and wrist movements – stronger than the one required to guess keystrokes.

Extension to laptops

We discussed extension of CSAW to laptops in Section 6.7.3, but it is worth emphasizing because of the increase in ownership and use of laptops. Extending initial authentication to laptops is straightforward. The challenge is in extending continuous authentication, which is due the differences between the way people interact with the mouse and the touchpad. If we revise our continuous approach to verify the user from her typing interactions alone, as we just discussed, we could easily extend continuous authentication to laptops.

Evaluation outside laboratory

Our evaluation of CSAW was based on laboratory studies; to evaluate how well CSAW performs in an uncontrolled settings, it is essential to conduct user studies outside the laboratory. Smartwatches make it feasible to conduct a usability study for CSAW outside laboratory, but the energy consumption on smartwatches is an issue. CSAW uses motion sensors continuously to offer a seamless user authentication experience, but the motion sensor data when processed on the main processor of a smartwatch consumes lot of energy; optimizing CSAW’s architecture and protocol to make it energy efficient is another future direction that will help improve CSAW’s utility. Current fitness bands and smartwatches last a day (or more) on a single charge while using motion sensors because their use of motion sensor is optimized at the hardware – the way we envision CSAW would be implemented. Until smartwatch manufacturers offer an energy optimized solution for apps to access and use motion sensor data, CSAW cannot be used to perform authentications throughout the day.

We can, however, assess CSAW's performance for limited duration in a day, e.g., to conduct a user study for 3-4 hours of a day (or while the smartwatch battery lasts with continuous use of motion sensors); such a study would not provide the usability assessment for using CSAW an entire day, but nonetheless it would provide useful information to assess CSAW's performance in an uncontrolled settings.

Delegation

Access delegation is an important aspect of people's workflow, especially in workplaces such as hospitals where clinicians often need to delegate permissions to their subordinates and peers. Unfortunately, many authentication schemes do not support delegation. For lack of an easy way to delegate permissions, people share credentials. To support access delegation, one must a) set up access-control policies that allow delegation, and b) provide easy and intuitive ways for users to express their intent and scope of delegation. We believe CSAW could help address the latter problem. For instance, synchronized gestures (shaking hands together or tap from one wristband to another) or a richer interface through user's smartphones could be used to delegate permissions or revoke permissions.

Next generation mobile devices

In Chapter 7 we focused on smartphones. However, CSAW provides a foundation for authentication that can be leveraged in the next generation of mobile devices. There are many devices other than smartphones that would benefit from continuous authentication, limitation and delegation. For example, high-end cameras will continue to evolve, and CSAW can protect pictures or video inside a camera and/or limit functionality from individuals other than the owner. Similarly, there are handheld devices (such as package scanners)

that are commonly handed to untrusted individuals for a signature or for confirming orders, etc. CSAW can limit what the untrusted individual can do with the handheld device in an unobtrusive manner. In general, it is likely that among emerging mobile devices (e.g., medical devices), there will be many that have limited user interfaces without keyboards and that could benefit from authentication that relies on simple motion sensors used while wearing an associated wristband.



Authentication behavior user study

Here we list the questions we used in the pre- and post-study interviews of participants of the user study described in Chapter 4.

A.1 Pre-logging interview

We used the following questions as guide in our semi-structured interviews, before the participants began self-logging their authentication events. In addition to these questions, we

welcomed topics and discussions about authentication initiated by participants.

- What is your typical day, in terms of authentication events?
- What targets and authenticators do you use?
- What do you carry with you?
- How do you manage your passwords?
- How do you choose/create passwords?

A.2 Post-Logging interview

We used the following questions as guide in our semi-structured interviews with the participants, after they logged their authentication events for one week. In addition to these questions, we also probed participants about their authentication behavior, based on the logging data that we had at this point.

- What is your most favorite authenticator?
- What is your least favorite authenticator?
- Did you log all events?
- Were you more aware of the authentication events?
- Did you notice any patterns?
- How do you feel about authentication events? (Multiple choice question)
 1. I don't even notice them.
 2. I notice them, but they rarely bug me.

3. They bug me, but not too much.
4. They bug me and I'd like to avoid them.
5. They are extremely frustrating.

- How do you feel about passwords?
- Do you have any comments/suggestions about the study?

B

Desktop user studies

B.1 Interview questions

After the participants finished the experiment in which they perform total 30 attempts with three different methods, we asked them the following questions.

- Have you tried any authentication method other than passwords?
- What do you like/dislike about passwords?

- What did you like/dislike about the tap-based method?
- What did you like/dislike about the mouse-based method?

B.2 SUS Survey

We modified the SUS form [14] by changing the word ‘system’ to ‘method’, and removing a question that is irrelevant to the authentication methods we tested in our user study. After the interview, participants filled out the form for each of three authentication methods they tried in the user study, rating each question on a 5-point Likert scale (5 being “Strongly agree”).

1. I think that I would like to use this method frequently.
2. I found this method unnecessarily complex.
3. I thought this method was easy to use.
4. I think that I would need the support of a technical person to be able to use this method.
5. I thought there was too much inconsistency in this method.
6. I would imagine that most people would learn to use this method very quickly.
7. I found this method very cumbersome to use.
8. I felt very confident using the method.
9. I needed to learn a lot of things before I could get going with this method.

Bibliography

- [1] A. Adams and M. A. Sasse. Users are not the enemy. *Communications of the ACM*, 42(12):40–46, Dec. 1999. DOI [10.1145/322796.322806](https://doi.org/10.1145/322796.322806). Citation on pages 3 and 84.
- [2] A. Adams, M. A. Sasse, and P. Lunt. Making passwords secure and usable. In H. Thimbleby, B. O’Conaill, and P. J. Thomas, editors, *People and Computers XII*, pages 1–19. Springer London, Jan. 1997. DOI [10.1007/978-1-4471-3601-9_1](https://doi.org/10.1007/978-1-4471-3601-9_1). Citation on page 57.
- [3] P. S. Aleksic and A. K. Katsaggelos. Audio-visual biometrics. *Proceedings of the IEEE*, 94(11):2025–2044, Nov. 2006. DOI [10.1109/JPROC.2006.886017](https://doi.org/10.1109/JPROC.2006.886017). Citation on page 195.
- [4] Android Wear. Online at <http://www.android.com/wear/>, visited May 2015. Citation on page 15.
- [5] Face unlock on Android 4.0. Online at http://www.huffingtonpost.com/2011/10/19/face-unlock-ice-cream-sandwich_n_1020207.html, visited May 2016. Citation on page 195.

- [6] Apple iPhone 5S. Online at <http://www.apple.com/iphone-5s>, visited Mar. 2016. Citation on page 26.
- [7] Apple Watch. Online at <https://www.apple.com/watch/>, visited Feb. 2016. Citation on pages 6, 15, 16, and 144.
- [8] D. Balzarotti, M. Cova, and G. Vigna. ClearShot: Eavesdropping on keyboard input from video. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, pages 170–183, May 2008. DOI [10.1109/SP.2008.28](https://doi.org/10.1109/SP.2008.28). Citation on page 186.
- [9] K. Baxter, C. Courage, and K. Caine. *Understanding Your Users: A Practical Guide to User Research Methods*. Morgan Kaufmann, second edition, 2015. Citation on page 113.
- [10] Biometric tech uses sound to distinguish ear cavity shape. Online at <https://www.helpnetsecurity.com/2016/03/10/biometric-tech-uses-sound-to-distinguish-ear-cavity-shape/>, visited Mar. 2016. Citation on page 26.
- [11] J. Blythe, R. Koppel, and S. W. Smith. Circumvention of security: Good users do bad things. *IEEE Security & Privacy*, 11(5):80–83, Sept. 2013. DOI [10.1109/MSP.2013.110](https://doi.org/10.1109/MSP.2013.110). Citation on page 85.
- [12] C. Bo, L. Zhang, X.-Y. Li, Q. Huang, and Y. Wang. SilentSense: Silent user identification via touch and movement behavioral biometrics. In *Proceedings of the International Conference on Mobile Computing and Networking (MobiCom)*, pages 187–190, 2013. DOI [10.1145/2500423.2504572](https://doi.org/10.1145/2500423.2504572). Citation on page 193.

- [13] J. Bonneau, C. Herley, P. C. van Oorschot, and F. Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *IEEE Symposium on Security and Privacy*, pages 553–567, May 2012. DOI [10.1109/SP.2012.44](https://doi.org/10.1109/SP.2012.44). Citation on pages 24, 182, and 194.
- [14] J. Brooke and Others. SUS – a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996. Citation on pages 113 and 206.
- [15] W. L. Bryan and N. Harter. Studies in the physiology and psychology of the telegraphic language. *Psychological Review*, 4(1):27–53, Jan. 1897. DOI <http://dx.doi.org/10.1037/h0073806>. Citation on page 26.
- [16] C. Cornelius and D. Kotz. Recognizing whether sensors are on the same body. *Journal of Pervasive and Mobile Computing*, 8(6):822–836, Dec. 2012. DOI [10.1016/j.pmcj.2012.06.005](https://doi.org/10.1016/j.pmcj.2012.06.005). Citation on pages 81 and 126.
- [17] C. Cornelius, R. Peterson, J. Skinner, R. Halter, and D. Kotz. A wearable system that knows who wears it. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pages 55–67, June 2014. DOI [10.1145/2594368.2594369](https://doi.org/10.1145/2594368.2594369). Citation on pages 6, 16, 26, 144, and 191.
- [18] M. D. Corner and B. Noble. Zero-interaction authentication. In *Proceedings of the International Conference on Mobile Computing and Networking (MobiCom)*, pages 1–11, 2002. DOI [10.1145/570645.570647](https://doi.org/10.1145/570645.570647). Citation on pages 25 and 187.
- [19] M. D. Corner and B. D. Noble. Protecting applications with transient authentication. In *Proceedings of the International Conference on Mobile Systems, Applications, and*

- Services (MobiSys)*, pages 57–70, 2003. DOI [10.1145/1066116.1066117](https://doi.org/10.1145/1066116.1066117). Citation on page 195.
- [20] L. F. Cranor. What’s wrong with your pa\$\$w0rd? Online at https://www.ted.com/talks/lorrie_faith_cranor_what_s_wrong_with_your_pa_w0rd, visited Mar. 2016. Citation on pages 25, 29, and 33.
- [21] L. F. Cranor and S. Garfinkel. *Security and usability: Designing secure systems that people can use*. O’Reilly Media Inc., 2005. Citation on page 33.
- [22] S. R. de Lima e Silva Filho and M. Roisenberg. Continuous authentication by keystroke dynamics using committee machines. In *IEEE International Conference on Intelligence and Security Informatics*, pages 686–687, 2006. DOI [10.1007/11760146_90](https://doi.org/10.1007/11760146_90). Citation on page 195.
- [23] B. Dosono, J. Hayes, and Y. Wang. “I’m stuck!”: A contextual inquiry of people with visual impairments in authentication. In *Proceedings of the Symposium On Usable Privacy and Security (SOUPS)*, pages 151–168, July 2015. Online at <https://www.usenix.org/conference/soups2015/proceedings/presentation/dosono>. Citation on pages 186 and 189.
- [24] S. Egelman, S. Jain, R. S. Portnoff, K. Liao, S. Consolvo, and D. Wagner. Are you ready to lock? In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 750–761, 2014. DOI [10.1145/2660267.2660273](https://doi.org/10.1145/2660267.2660273). Citation on page 32.
- [25] Enron email dataset. Online at <http://www.cs.cmu.edu/~enron/>, visited Mar. 2016. Citation on page 166.

- [26] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006. DOI [10.1016/j.patrec.2005.10.010](https://doi.org/10.1016/j.patrec.2005.10.010). Citation on page 20.
- [27] Fitbit. Online at <http://www.fitbit.com>, visited May 2015. Citation on page 15.
- [28] D. Florencio and C. Herley. A large-scale study of web password habits. In *Proceedings of the International Conference on World Wide Web (WWW)*, pages 657–666, 2007. DOI [10.1145/1242572.1242661](https://doi.org/10.1145/1242572.1242661). Citation on page 33.
- [29] ATAP’s project Abacus aims to eliminate passwords from our lives. Online at <http://www.androidauthority.com/google-atap-project-abacus-eliminates-passwords-612678/>, visited Mar. 2016. Citation on page 27.
- [30] A. Greenberg. The app I used to break into my neighbor’s home. Online at <http://www.wired.com/2014/07/keyme-let-me-break-in/>, visited Mar. 2016. Citation on page 29.
- [31] M. Harbach, E. von Zezschwitz, A. Fichtner, A. D. Luca, and M. Smith. It’s a hard lock life: A field study of smartphone (un)locking behavior and risk perception. In *Proceedings of the Symposium On Usable Privacy and Security (SOUPS)*, pages 213–230, July 2014. Online at <https://www.usenix.org/conference/soups2014/proceedings/presentation/harbach>. Citation on page 51.
- [32] E. Hayashi and J. Hong. A diary study of password usage in daily life. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI)*, pages 2627–2630, May 2011. DOI [10.1145/1978942.1979326](https://doi.org/10.1145/1978942.1979326). Citation on page 33.

- [33] C. Herley. So long, and no thanks for the externalities: The rational rejection of security advice by users. In *Proceedings of the Workshop on New Security Paradigms Workshop (NSPW)*, pages 133–144, 2009. DOI [10.1145/1719030.1719050](https://doi.org/10.1145/1719030.1719050). Citation on page 25.
- [34] D. Hintze, R. D. Findling, M. Muaaz, S. Scholz, and R. Mayrhofer. Diversity in locked and unlocked mobile device usage. In *Proceedings of the ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*, pages 379–384, Sept. 2014. DOI [10.1145/2638728.2641697](https://doi.org/10.1145/2638728.2641697). Citation on page 51.
- [35] O. Huhta, S. Udar, M. Juuti, P. Shrestha, N. Saxena, and N. Asokan. Pitfalls in designing zero-effort deauthentication: Opportunistic human observation attacks. In *Proceedings of the Network and Distributed Systems Security Symposium (NDSS)*, Feb. 2016. DOI [10.14722/ndss.2016.23199](https://doi.org/10.14722/ndss.2016.23199). Citation on page 144.
- [36] I. Ion, R. Reeder, and S. Consolvo. “...no one can hack my mind”: Comparing expert and non-expert security practices. In *Proceedings of the Symposium On Usable Privacy and Security (SOUPS)*, pages 327–346, 2015. Online at <https://www.usenix.org/system/files/conference/soups2015/soups15-paper-ion.pdf>. Citation on pages 29, 33, and 59.
- [37] A. K. Jain, A. Ross, and S. Pankanti. Biometrics: A tool for information security. *IEEE Transactions on Information Forensics and Security*, 1(2):125–143, June 2006. DOI [10.1109/TIFS.2006.873653](https://doi.org/10.1109/TIFS.2006.873653). Citation on page 26.
- [38] Jawbone UP. Online at <https://jawbone.com/up>, visited Dec. 2015. Citation on page 15.

- [39] A. Kalamandeen, A. Scannell, E. de Lara, A. N. Sheth, and A. LaMarca. Ensemble: Cooperative proximity-based authentication. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pages 331–344, June 2010. DOI [10.1145/1814433.1814466](https://doi.org/10.1145/1814433.1814466). Citation on page 82.
- [40] R. Koppel, J. P. Metlay, A. Cohen, B. Abaluck, A. R. Localio, S. E. Kimmel, and B. L. Strom. Role of computerized physician order entry systems in facilitating medication errors. *JAMA: The Journal of the American Medical Association*, 293(10):1197–1203, 2005. DOI [doi:10.1001/jama.293.10.1197](https://doi.org/10.1001/jama.293.10.1197). Citation on page 84.
- [41] J. Krumm and D. Rouhana. Placer: Semantic place labels from diary data. In *Proceedings of the International Conference on Ubiquitous Computing (UbiComp)*, pages 163–172, Sept. 2013. DOI [10.1145/2493432.2493504](https://doi.org/10.1145/2493432.2493504). Citation on page 47.
- [42] A. Kumar, N. Saxena, G. Tsudik, and E. Uzun. A comparative study of secure device pairing methods. *Pervasive and Mobile Computing*, 5(6):734–749, 2009. DOI [10.1016/j.pmcj.2009.07.008](https://doi.org/10.1016/j.pmcj.2009.07.008). Citation on page 15.
- [43] C. E. Landwehr. Protecting unattended computers without software. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, pages 274–285, 1997. Online at <http://dl.acm.org/citation.cfm?id=872015.872112>. Citation on pages 25, 188, and 195.
- [44] D. Lee. Keyless cars ‘increasingly targeted by thieves using computers’. Online at <http://www.bbc.com/news/technology-29786320>, visited Mar. 2016. Citation on page 34.

- [45] L. Li, X. Zhao, and G. Xue. Unobservable re-authentication for smartphones. In *Proceedings of the Network and Distributed Systems Security Symposium (NDSS)*, 2012. Citation on pages 27, 193, and 195.
- [46] I. T. L. Lillegaard, E. B. Løken, and L. F. Andersen. Relative validation of a pre-coded food diary among children, under-reporting varies with reporting day and time of the day. *European Journal of Clinical Nutrition*, 61(1):61–68, 2007. DOI [10.1038/sj.ejcn.1602487](https://doi.org/10.1038/sj.ejcn.1602487). Citation on pages 31 and 37.
- [47] J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan. uWave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing*, 5(6):657–675, Dec. 2009. DOI [10.1016/j.pmcj.2009.07.007](https://doi.org/10.1016/j.pmcj.2009.07.007). Citation on page 155.
- [48] X. Liu, Z. Zhou, W. Diao, Z. Li, and K. Zhang. When good becomes evil: Keystroke inference with smartwatch. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 1273–1285, 2015. DOI [10.1145/2810103.2813668](https://doi.org/10.1145/2810103.2813668). Citation on pages 146 and 198.
- [49] Lookout. Mobile mindset study. Online at <https://www.lookout.com/resources/reports/mobile-mindset>, visited June 2012. Citation on page 2.
- [50] Survey reveals consumers exhibit risky behaviors despite valuing their privacy on mobile devices. Online at <https://www.lookout.com/news-mobile-security/sprint-lookout-mobile-privacy-survey>, visited May 2016. Citation on pages 32 and 151.

- [51] S. Mare, M. Baker, and J. Gummesson. A study of authentication in daily life. In *Proceedings of the Symposium On Usable Privacy and Security (SOUPS)*, June 2016. Online at <https://www.usenix.org/conference/soups2016/technical-sessions/presentation/mare>. Citation on page 7.
- [52] S. Mare, A. Molina-Markham, C. Cornelius, R. Peterson, and D. Kotz. ZEBRA: Zero-effort bilateral recurring authentication. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, pages 705–720, May 2014. This project has been renamed BRACE, DOI [10.1109/SP.2014.51](https://doi.org/10.1109/SP.2014.51). Citation on page 7.
- [53] P. Marquardt, A. Verma, H. Carter, and P. Traynor. (sp)iPhone: Decoding Vibrations From Nearby Keyboards Using Mobile Phone Accelerometers. In *Proceedings of 18th ACM Conference on Computer and Communications Security (CCS)*, pages 551–562, Oct. 2011. DOI [10.1145/2046707.2046771](https://doi.org/10.1145/2046707.2046771). Citation on pages 184 and 192.
- [54] R. Mayrhofer and H. Gellersen. Shake well before use: Intuitive and secure pairing of mobile devices. *IEEE Transactions on Mobile Computing*, 8(6):792–806, June 2009. DOI [10.1109/TMC.2009.51](https://doi.org/10.1109/TMC.2009.51). Citation on page 82.
- [55] T. C. Meng, P. Gupta, and D. Gao. I can be you: Questioning the use of keystroke dynamics as biometrics. In *Proceedings of the Network and Distributed Systems Security Symposium (NDSS)*, 2013. Online at <http://flyer.sis.smu.edu.sg/ndss13-tey.pdf>. Citation on page 192.
- [56] M. B. Miles and A. M. Huberman. *Qualitative data analysis: An expanded sourcebook*. Sage, second edition, 1994. Citation on page 48.
- [57] Misfit. Online at <http://misfit.com>, visited May 2015. Citation on page 15.

- [58] F. Monroe and A. D. Rubin. Keystroke dynamics as a biometric for authentication. *Future Generation Computer Systems*, 16(4):351–359, 2000. DOI [10.1016/S0167-739X\(99\)00059-X](https://doi.org/10.1016/S0167-739X(99)00059-X). Citation on page 27.
- [59] R. Morris and K. Thompson. Password security: A case history. *Communications of the ACM (CACM)*, 22(11):594–597, Nov. 1979. DOI [10.1145/359168.359172](https://doi.org/10.1145/359168.359172). Citation on page 24.
- [60] Motorola. Motorola MOTOACTV. Online at <https://motoactv.com/home/page/features.html>, visited Mar. 2016. Citation on pages 30 and 38.
- [61] K. Mowery, S. Meiklejohn, and S. Savage. Heat of the moment: Characterizing the efficacy of thermal camera-based attacks. In *Proceedings of the USENIX Workshop on Offensive Technologies (WOOT)*, Aug. 2011. Online at http://www.usenix.org/events/woot11/tech/final_files/Mowery.pdf. Citation on pages 177 and 184.
- [62] A. Narayanan, N. Thiagarajan, M. Lakhani, M. Hamburg, and D. Boneh. Location privacy via private proximity testing. In *Proceedings of the Network and Distributed Systems Security Symposium (NDSS)*, Feb. 2011. Online at <http://www.internetsociety.org/sites/default/files/nara.pdf>. Citation on page 82.
- [63] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970. DOI [10.1016/0022-2836\(70\)90057-4](https://doi.org/10.1016/0022-2836(70)90057-4). Citation on page 99.
- [64] NIST/SEMATECH. *e-Handbook of Statistical Methods*. NIST, Apr. 2012. Online at <http://www.itl.nist.gov/div898/handbook/>. Citation on page 126.

- [65] E. Oliver. The challenges in large-scale smartphone user studies. In *Proceedings of the ACM International Workshop on Hot Topics in Planet-scale Measurement (HotPlanet)*, June 2010. DOI [10.1145/1834616.1834623](https://doi.org/10.1145/1834616.1834623). Citation on page 2.
- [66] A. Peacock, X. Ke, and M. Wilkerson. Typing patterns: A key to user identification. *IEEE Security & Privacy*, 2(5):40–47, 2004. DOI <http://doi.ieeecomputersociety.org/10.1109/MSP.2004.89>. Citation on page 27.
- [67] Pebble. Online at <https://getpebble.com>, visited May 2015. Citation on page 15.
- [68] S. Peisert, E. Talbot, and T. Kroeger. Principles of authentication. In *Proceedings of the Workshop on New Security Paradigms Workshop (NSPW)*, pages 47–56, Sept. 2013. DOI [10.1145/2535813.2535819](https://doi.org/10.1145/2535813.2535819). Citation on page 85.
- [69] Device ownership over time. Pew Research Center, Online at <http://www.pewinternet.org/data-trend/mobile/device-ownership/>, visited Feb. 2016. Citation on page 2.
- [70] M.-Z. Poh, D. J. McDuff, and R. W. Picard. Non-contact, automated cardiac pulse measurements using video imaging and blind source separation. *Optics Express*, 18(10):10762–10774, May 2010. DOI [10.1364/oe.18.010762](https://doi.org/10.1364/oe.18.010762). Citation on page 80.
- [71] Polybius. Histories. Perseus project, Tufts University, Online at <http://www.perseus.tufts.edu/hopper/text?doc=urn:cts:greekLit:tlg0543.tlg001.perseus-eng1:1.45>, visited Mar. 2016. Citation on page 24.
- [72] D. M. Powers. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Research*, 2(1):37–63, 2011. Online at <http://dspace.flinders.edu.au/xmlui/handle/2328/27165>. Citation on page 20.

- [73] R. Raguram, A. M. White, D. Goswami, F. Monrose, and J.-M. Frahm. iSpy: Automatic reconstruction of typed input from compromising reflections. In *Proceedings of ACM conference on Computer and Communications Security (CCS)*, pages 527–536, 2011. DOI [10.1145/2046707.2046769](https://doi.org/10.1145/2046707.2046769). Citation on pages 147 and 184.
- [74] K. B. Rasmussen, M. Roeschlin, I. Martinovic, and G. Tsudik. Authentication using pulse-response biometrics. In *Proceedings of the Network and Distributed Systems Security Symposium (NDSS)*, Feb. 2014. Online at http://www.internetsociety.org/sites/default/files/08_1_0.pdf. Citation on pages 185, 191, and 195.
- [75] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman. Activity recognition from accelerometer data. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1541–1546, 2005. Online at <http://portal.acm.org/citation.cfm?id=1620107>. Citation on page 126.
- [76] Human reaction time test. Online at <http://www.humanbenchmark.com/tests/reactiontime/>, visited Feb. 2016. Citation on page 90.
- [77] A. Ross, J. Shah, and A. K. Jain. From template to image: Reconstructing fingerprints from minutiae points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(4):544–560, Apr. 2007. DOI [10.1109/TPAMI.2007.1018](https://doi.org/10.1109/TPAMI.2007.1018). Citation on page 195.
- [78] RSA. RSA SecurID two-factor authentication. Online at <https://www.rsa.com/en-us/products-services/identity-access-management/securid>, visited Mar. 2016. Citation on page 25.

- [79] A. L. Scherr. *An analysis of time-shared computer systems*. PhD thesis, Massachusetts Institute of Technology (MIT), 1967. PhD Thesis, Online at <ftp://129.69.211.95/pdf/mit/lcs/tr/MIT-LCS-TR-018.pdf>. Citation on page 24.
- [80] A. Serwadda and V. V. Phoha. When kids’ toys breach mobile phone security. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 599–610, 2013. DOI [10.1145/2508859.2516659](https://doi.org/10.1145/2508859.2516659). Citation on pages 147 and 193.
- [81] G. Shah, A. Molina, and M. Blaze. Keyboards and covert channels. In *Proceedings of the USENIX Security Symposium (USENIX Security)*, 2006. Online at <http://static.usenix.org/events/sec06/tech/shah/shah.pdf>. Citation on page 148.
- [82] Shimmer Research. Online at <http://www.shimmersensing.com>, visited Feb. 2016. Citation on pages 102 and 167.
- [83] D. Shukla, R. Kumar, A. Serwadda, and V. V. Phoha. Beware, your hands reveal your secrets! In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 904–917, 2014. DOI [10.1145/2660267.2660360](https://doi.org/10.1145/2660267.2660360). Citation on page 177.
- [84] S. Sinclair. *Access Control In and For the Real World*. PhD thesis, Dartmouth Computer Science, Nov. 2013. Online at <http://www.cs.dartmouth.edu/reports/abstracts/TR2013-745/>. Citation on page 79.
- [85] S. Sinclair and S. W. Smith. What’s wrong with access control in the real world? *IEEE Security and Privacy*, 8(4):74–77, July 2010. DOI [10.1109/MSP.2010.139](https://doi.org/10.1109/MSP.2010.139). Citation on pages 3 and 85.

- [86] S. Sinclair and S. W. Smith. Access control realities as observed in a clinical medical setting. Technical Report TR2012-714, Dartmouth College, Computer Science, Jan. 2012. Online at http://www.cs.dartmouth.edu/cms_file/SYS_techReport/582/TR2012-714.pdf. Citation on pages 3 and 191.
- [87] S. W. Smith and R. Koppel. Healthcare information technology’s relativity problems: A typology of how patients’ physical reality, clinicians’ mental models, and healthcare information technology differ. *Journal of the American Medical Informatics Association*, 2013. DOI [10.1136/amiajnl-2012-001419](https://doi.org/10.1136/amiajnl-2012-001419). Citation on page 85.
- [88] J. C. Spender. Identifying computer users with authentication devices (tokens). *Computers & Security*, 6(5):385–395, 1987. DOI [10.1016/0167-4048\(87\)90011-3](https://doi.org/10.1016/0167-4048(87)90011-3). Citation on page 25.
- [89] S. Sridhar, P. Misra, and J. Warrior. CheepSync: A time synchronization service for resource constrained Bluetooth low energy advertisers. *CoRR*, abs/1501.06479, Jan. 2015. Online at <http://arxiv.org/abs/1501.06479>. Citation on page 16.
- [90] F. Stajano. Pico: No more passwords! In B. Christianson, B. Crispo, J. Malcolm, and F. Stajano, editors, *Security Protocols XIX*, volume 7114 of *Lecture Notes in Computer Science*, pages 49–81. Springer-Verlag Berlin, Mar. 2011. DOI [10.1007/978-3-642-25867-1_6](https://doi.org/10.1007/978-3-642-25867-1_6). Citation on pages 25, 144, 186, and 195.
- [91] M. Steves, D. Chisnell, A. Sasse, K. Krol, M. Theofanos, and H. Wald. Report: Authentication diary study. Technical Report NISTIR 7983, National Institute of Standards and Technology (NIST), 2014. DOI [10.6028/NIST.IR.7983](https://doi.org/10.6028/NIST.IR.7983). Citation on page 32.

- [92] E. Stobert and R. Biddle. The password life cycle: User behaviour in managing passwords. In *Proceedings of the Symposium On Usable Privacy and Security (SOUPS)*, pages 243–255, July 2014. Online at <https://www.usenix.org/conference/soups2014/proceedings/presentation/stobert>. Citation on page 59.
- [93] F. Tari, A. A. Ozok, and S. H. Holden. A comparison of perceived and real shoulder-surfing risks between alphanumeric and graphical passwords. In *Proceedings of the Symposium On Usable Privacy and Security (SOUPS)*, pages 56–66, 2006. DOI [10.1145/1143120.1143128](https://doi.org/10.1145/1143120.1143128). Citation on page 186.
- [94] S. Trewin, C. Swart, L. Koved, J. Martino, K. Singh, and S. Ben-David. Biometric authentication on a mobile device: A study of user effort, error and task disruption. In *Proceedings of the Annual Computer Security Applications Conference*, pages 159–168, Dec. 2012. DOI [10.1145/2420950.2420976](https://doi.org/10.1145/2420950.2420976). Citation on page 84.
- [95] D. Umphress and G. Williams. Identity verification through keyboard characteristics. *International Journal of Man-Machine Studies*, 23(3):263–273, 1985. DOI [http://dx.doi.org/10.1016/S0020-7373\(85\)80036-5](http://dx.doi.org/10.1016/S0020-7373(85)80036-5). Citation on page 26.
- [96] I. Urbina. The secret life of passwords. Online at <http://www.nytimes.com/2014/11/19/magazine/the-secret-life-of-passwords.html>, visited Feb. 2016. Citation on page 33.
- [97] D. Van Bruggen, S. Liu, M. Kajzer, A. Striegel, C. R. Crowell, and J. D’Arcy. Modifying smartphone user locking behavior. In *Proceedings of the Symposium On Usable Privacy and Security (SOUPS)*, 2013. DOI [10.1145/2501604.2501614](https://doi.org/10.1145/2501604.2501614). Citation on page 32.

- [98] D. Wagner, A. Rice, and A. Beresford. Device analyzer: Understanding smartphone usage. In *Proceedings of the International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services (MobiQuitous)*, pages 195–208, Dec. 2013. DOI [10.1007/978-3-319-11569-6_16](https://doi.org/10.1007/978-3-319-11569-6_16). Citation on pages 2 and 51.
- [99] H. Wang, T. T. Lai, and R. R. Choudhury. Mole: Motion leaks through smartwatch sensors. In *Proceedings of the International Conference on Mobile Computing and Networking (MobiCom)*, 2015. Online at <http://web.engr.illinois.edu/~hewang5/papers/mole-final.pdf>. Citation on pages 146 and 198.
- [100] R. Witty and K. Brittain. Password reset: Self-service that you will love, Mar. 2016. Online at <https://www.gartner.com/doc/354760/password-reset-selfservice-love>. Citation on page 29.
- [101] Yubikey. Online at <https://www.yubico.com>, visited Mar. 2016. Citation on page 25.
- [102] Q. Yue, Z. Ling, X. Fu, B. Liu, K. Ren, and W. Zhao. Blind recognition of touched keys on mobile devices. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, pages 1403–1414, 2014. DOI [10.1145/2660267.2660288](https://doi.org/10.1145/2660267.2660288). Citation on page 177.
- [103] N. Zheng, K. Bai, H. Huang, and H. Wang. You are how you touch: User verification on smartphones via tapping behaviors. In *Proceedings of the IEEE International Conference on Network Protocols (ICNP)*, pages 221–232, Oct. 2014. DOI [10.1109/ICNP.2014.43](https://doi.org/10.1109/ICNP.2014.43). Citation on page 27.